

COLECȚIA INFORMATICA



Octavian Căpăţînă

PROIECTAREA CU MICROCALCULATOARE INTEGRATE

EDITURA DACIA



COLECȚIA INFORMATICA



ULTIMA PROVOCARE A MICROELECTRONICII ȘI UN ÎNDEMN

Cheile recunoscute ale dezvoltării, în acest sfîrșit de secol, sînt microelectronica și informatica. Generalizarea acestora, în sensul propriu al cuvîntului, este limitată de un prag senzorial. Senzorii sînt cealaltă cheie a dezvoltării. Numai cînd parametrii procesului, mașinii, ambiantului vor putea fi ușor și ieftin evaluați destinul microelectronicii și informaticii se va împlini. Cred că trebuie să recuperăm decalajul în domeniul senzorilor prin asamblarea rapidă a unor colective de dezvoltare și microproducție pe lîngă marile firme din domeniul microelectronicii, cu sprijinul Departamentului Științei, și prin înființarea unei facultăți de profil.

OCTAVIAN CĂPĂȚÎNĂ

Lei 426,–

ISBN 973-35-0309-6

Coperta colecției M. BACIU

OCTAVIAN CĂPĂȚÎNĂ

PROIECTAREA CU MICROCALCULATOARE INTEGRATE

EDITURA DACIA, CLUJ-NAPOCA 1992

Redactor: MONICA CREMENE
Tehnoredactor: CONSTANTIN RUSU
Corector: FELICIA SCHLEZAK

Apărut: 1993. Bun de tipar: 28. 12. 1992. Comanda nr. 3345.
Coli de tipar: 10,75. Tiraj: 7000 + 100 + 10 ex. broşate.
Hîrtie: velină 70 g/mp. Format: 61 × 86/16.

Tiparul executat sub comanda nr. 370
la Imprimeria „ARDEALUL" Cluj-Napoca,
B-dul 22 Decembrie nr. 146.
ROMÂNIA



ISBN 973-35-0309-6

PREFAŢĂ

Caracterul intelectual al societăţii spunea Spiru Haret este dat de nivelul inteligenţei medii a societăţii. Bunăstarea socială, la rîndul ei e determinată de această inteligenţă medie a societăţii. Tot un om de ştiinţă român arăta că: „Ştiinţa, literatura dar şi tehnologia în special tehnologia utilizării calculatoarelor electronice şi a inteligenţei artificiale pot contribui la creşterea inteligenţei sociale.".*

Este deci unul din imperativele actuale alinierea industriei noastre la avansul fulminant al ştiinţei şi tehnologiei actuale, îndeosebi al microelectronicii. Progresele acestei ramuri tehnice au permis plasarea de suficiente resurse pe o aşchie de Si pentru a realiza un adevărat calculator într-o singură pastilă; pastilă care prin programare poate dezvolta orice funcţii logice necesare unor sisteme digitale, mergînd de la supravegherea unui automobil sau comanda unei maşini de spălat pînă la ghidarea unei rachete.

Microelectronica înglobată în produse numită şi electronica funcţională, duce la reducerea substanţială a componentelor mecanice. Exemplele sînt numeroase şi edificatoare: ceasul electronic, plotterul cu rastru, maşina de cusut etc. Prin înglobarea de informaţie, de inteligenţă în produse scade partea materială. Deci este pe deplin justificată aserţiunea conform căreia microelectronica este nu numai o tehnologie de vîrf ci şi o resursă economică.

În acest context general se înscrie şi efortul de a prezenta proiectarea de sisteme logice cu microcalculatoare integrate.

17. 04. 89 Cluj-Napoca, *AUTORUL*

* Mihai Drăgănescu Informatica şi societatea, Ed. politică, 1988.

CUPRINS

1. INTRODUCERE

1.1. UN NOU MOD DE A PRIVI PROIECTAREA SISTEMELOR LOGICE

Din 1957, anul apariției tranzistorului planar, și pînă astăzi numărul de componente „integrate" pe o așchie plană de SI s-a dublat în fiecare an. În tot acest răstimp prețul circuitului integrat s-a menținut același dacă se ia în considerare și inflația [4]. Prin urmare dacă luăm în considerare prețul pe componenta „integrată", acesta a scăzut la $1/2^{**}$ (1984—1964), adică de 1 milion de ori. Procesul de fabricație a circuitelor integrate este răspunzător de acest ritm fără precedent în istoria tehnicii. Pe o plachetă de Si cu diametrul de 3 țoli (aproximativ 4500 mm^2) se procesează simultan mii de circuite integrate identice, fiecare circuit ocupînd o suprafață de 2—3 mm^2. Fiecare etapă a procesului de fabricație se realizează pe cîte o plachetă sau chiar pe mai multe odată, deci pentru mii de circuite simultan. Numai încapsularea și testarea finală se face individual, în rest, efortul de a realiza un circuit integrat sau o mie de circuite integrate fiind același. Faptul că un circuit SSI are aproximativ același preț cu un circuit MSI sau LSI se justifică tocmai prin ceea ce s-a subliniat mai sus. Prețul de livrare a circuitelor LSI este, totuși, mult mai mare în primii ani de viață ai circuitului. Această diferență se datorează mai multor factori între care și:

— recuperarea cheltuielilor de proiectare-dezvoltare a noului circuit LSI,

— procentului mai mic de reușită datorită punerii la punct insuficiente a procesului tehnologic (de obicei nou),

— costului mai mare al testării.

Se observă că primii doi factori sînt temporari. Odată cu amortizarea cheltuielilor de proiectare-dezvoltare și cu stăpînirea noului proces tehnologic prețul circuitului LSI scade spre nivelul prețului de producție adică la nivelul prețului circuitelor SSI și MSI. Din

păcate în cazul nostru, particular, prețul circuitelor integrate este ridicol de mare, „bătut în cuie" pentru mulți ani. Amortizarea cheltuielilor de asimilare, creșterea procentului de reușită nu au nici o influență asupra prețului de livrare. Acest fapt se manifestă cît se poate de negativ în efortul de modernizare a producției, a creșterii productivității muncii și în general asupra progresului tehnic.

Rata de defectare a unui circuit LSI este aproape aceeași cu rata de defectare a unui circuit SSI. Defectele intrinseci, pentru că la acestea ne referim, se datorează încapsulării și interconexiunilor interne. Dacă ținem cont că un circuit LSI înlocuiește zeci pînă la sute de circuite SSI, atunci realizarea unei funcții cu LSI va avea o rată de defectare de zeci pînă la sute de ori mai mică decît aceeași funcție implementată cu circuite SSI.

Standardizarea și producția de serie au condus la realizarea unor producții de masă, concretizate într-o bogăție ce tinde să satisfacă cerințele individuale într-o perioadă de explozie demografică. Circuitele integrate, a căror producție se realizează în serii foarte mari, sînt de fapt niște produse standardizate. Logica binară și ca atare circuitele digitale se pretează cel mai bine la standardizare datorită simplității. Prin circuitele digitale s-a atins o culme a standardizării care nu poate a fi amenințată de nimic [4]. Componentele mecanice nu pot atinge gradul de standardizare al circuitelor integrate în primul rînd datorită seriilor mult mai mici. Consumul de materii prime și de energie din cazul componentelor mecanice vor păstra și adînci perpetuu acest decalaj față de circuitele integrate.

Un exemplu edificator și la îndemîna fiecăruia ni-l oferă orologia. Oricît de automatizată ar fi astăzi producerea unui ceas mecanic, prețul de cost al acestuia este de cel puțin 10 ori mai mare decît prețul unui ceas electronic. Dacă prețul unui ceas cu roți dințate ar fi să zicem 500 lei, atunci prețul unui ceas electronic ar trebui să fie cel mult 50 lei. Astăzi nu se mai fac investiții pentru producția de ceasuri mecanice, iar industria elvețiană de ceasuri în prag de faliment s-a salvat tocmai printr-o foarte rapidă adaptare la producția ceasurilor electronice. În concluzie ori de cîte ori se proiectează un sistem trebuie avut în vedere ca obiectiv major înlocuirea pe cît posibil a componentelor mecanice cu componente electronice. Și tendința de creștere a siguranței în funcționare pledează pentru aceeași concluzie. Rata de defectare a circuitelor integrate este foarte mică. Circuitele integrate puse la punct și bine testate de la firme cu pretenții au timpul mediu de defectare, MTBF, de zeci și chiar sute de ani. Defectarea circuitelor integrate se datorează mai ales utilizării lor necorespunzătoare din punct de vedere electric și/sau termic. Totuși ponderea circuitelor integrate în prețul de cost al unui sistem este mai mică de 10% [4] și tinde

să scadă în continuare. În timp ce evoluția circuitelor integrate a cunoscut ritmuri fără precedent în ceea ce privește realizarea de funcții logice complexe în condiții de siguranță mare și la un preț tot mai mic, nu se poate spune același lucru despre:

— sursele de alimentare,
— părțile mecanice de asamblare,
— întreținerea și service-ul

care dețin ponderea cea mai mare în costul unui sistem. O judecată simplă dar și simplistă ne conduce la concluzia că pentru a reduce prețul unui sistem trebuie să acționăm asupra părților mecanice aferente și surselor de alimentare. Acestea dețin o pondere de aproximativ 90% din costul total al unui sistem logic. Îmbunătățirile previzibile din sectorul mecanic vor fi și în continuare lente, ca și pînă acum, ca atare nu vor duce la o scădere a prețului comparabilă cu cea a componentelor integrate. Și atunci calea cea mai eficace de a scădea prețul unui sistem este aceea de a reduce tocmai numărul de circuite integrate. Reducerea drastică a circuitelor integrate este posibilă tocmai prin apariția circuitelor LSI și în special prin apariția microcalculatoarelor integrate. De exemplu dacă se reduce numărul circuitelor integrate ale unui sistem logic la un sfert prin utilizarea circuitelor larg integrate se reduce ca urmare la un sfert și:

— numărul de plăci de circuit imprimat,
— numărul de conectoare,
— numărul de sertare,
— numărul de rack-uri
— volumul și prețul surselor de alimentare,
— cheltuielile de proiectare, întreținere și service.

Așa stînd lucrurile, obiectivele proiectării tradiționale care urmărea reducerea numărului de porți și bistabile pe cale logică (diagrame Karnaugh, tabele de adevăr etc.) nu mai corespund etapei actuale a proiectării cu circuite larg integrate. Are loc o translație a obiectivelor proiectării sistemelor logice de la reducerea numărului de porți și bistabile la reducerea numărului de capsule integrate. De multe ori, prin utilizarea unor circuite LSI în locul unei implementări cu circuite SSI numărul de porți și bistabile „integrate" crește de fapt; cu toate acestea prețul de ansamblu va scădea. Thomas Blakeslee în [4] face analiza costurilor dintr-un sistem digital și ajunge la concluzia că prețul total este proporțional cu numărul de capsule integrate din sistem; factorul de proporționalitate fiind de

3,31 $/capsulă. Costul mediu al unui circuit integrat SSI, MSI sau LSI considerat în respectiva analiză de 0,4$, preț ce corespunde realităților de acum un deceniu. Ținînd cont și de rata inflației, permanentă, putem considera, în continuare, un preț de 0,3 $ valabil astăzi. Reducînd, de exemplu, 30 de circuite SSI printr-un circuit LSI obținem o reducere a costurilor de producție de:

30 caps. * 0,3 $/caps. + 30 caps. * 3,31 $ cost supl./caps.
minus
1 caps. * 1,0 $/caps. + 1 caps * 3,31 $ cost. supl./caps.
adică de 103,99 $.

Am considerat prețul mediu al circuitelor SSI înlocuite 0,3 $/buc. și prețul circuitului LSI de 1 $. Se observă că reducînd 30 de capsule SSI care costă 9 $ se obține o reducere a costului total al sistemului de 103 $ care este echivalentul a 300 de circuite SSI sau a 100 de circuite LSI.

În tehnicile de proiectare, pe care de acum le vom considera tradiționale funcțiile logice se obțin prin interconectarea porților logice și a bistabilelor. Deoarece circuitele LSI cuprind suficiente elemente de circuit au trebuit găsite modalitățile de a configura din exterior aceste elemente integrate din interiorul circuitului LSI. Modalitățile la care ne referim se numesc microprogramare și programare. Configurarea elementelor interne se face prin program. Circuitele LSI a căror caracteristici logice sînt definite de către un program, sînt de fapt microprocesoare sau microcalculatoare integrate. Microcalculatoarele integrate sînt niște circuite LSI standard capabile să satisfacă nevoile unui cerc foarte larg de aplicații de la ghidarea unei rachete la controlul unei sobe cu raze infraroșii. Din acest punct de vedere circuitul LSI programabil care are înglobate memorie și circuite de intrare/ieșire, microcalculatorul integrat deci, se poate considera ca o culme a circuitelor LSI și VLSI. O culme pentru că acest circuit programabil este universal, îl putem considera standard și poate funcționa de sine stătător.

*

* *

În concluzie putem considera:

I. la nivelul concepției unui nou produs (mecanică + electronică și informatică) este importantă reducerea părții mecanice pe seama electronicii și informaticii. Altfel spus pe seama inteligenței care se înglobează în noul produs via informaticii prin microelectronică și

II. la nivelul părții electronice a noului produs este importantă reducerea numărului de circuite integrate pe seama folosirii microcalculatoarelor integrate și a microcontrolerelor integrate care prin program, pe de o parte suplinesc funcții logice altădată realizate prin logica cablată iar pe de altă parte adaugă noi funcții la cele vechi fără nici un consum material doar pe seama subprogramelor înglobate.

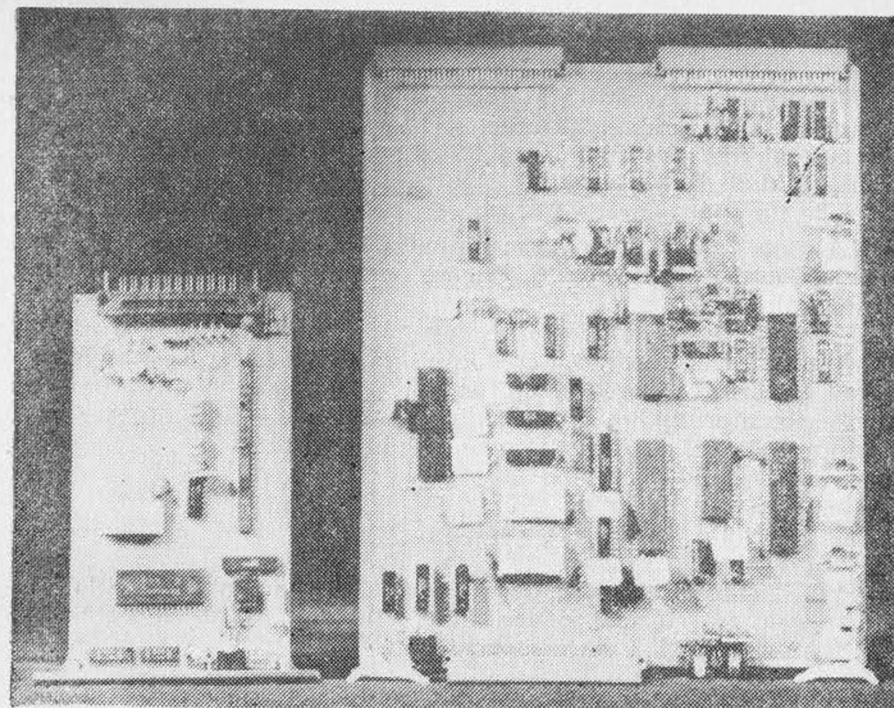


Fig. 1.0. Modul de control a unei axe de mișcare la un robot industrial realizat în două variante: cu microprocesor și cu microcalculator integrat.

Oferim în figura 1.0 o imagine concretă a pasului făcut trecînd de la un microprocesor, „de vîrf" de 8 biți de uz general, la un microcalculator integrat. Se văd două variante ale modului de control a mișcării pe o axă la un robot industrial [13] realizate una în 1986 cu microprocesorul Z80 și cealaltă în 1988 cu microcalculatorul integrat Z8. În acest caz concret s-a redus numărul circuitelor integrate de la 44, în cazul proiectului cu Z80, la 12, în aceeași proporție reducîndu-se și suprafața circuitului imprimat.

1.2. DOMENII DE UTILIZARE

Microcalculatorul integrat [one chip microcomputer, singel chip microcomputer, Einchipmikrorechnern] conține pe lîngă unitatea centrală — microprocesorul cunoscut — și: memorie citește/scrie (RAM), memorie fixă (ROM), canale de intrare/ieșire paralele și serie, circuite de timp, circuite de tratare a întreruperilor și oscilatorul pilot.

La toate microcalculatoarele integrate de pînă acum se observă un mare număr de intrări/ieșiri, care permit o interfață confortabilă cu mediul extern.

Prezența resurselor de interfațare, a memoriei interne și a oscilatorului pilot (mai puțin a cristalului de cuarț) pe aceeași așchie de siliciu asigură realizarea anumitor aplicații aproape exclusiv pe baza microcalculatorului integrat. Reducerea hardware-ului necesar unei anumite aplicații conduce la niște avantaje certe:

— volum mic,
— fiabilitate ridicată,
— consum de energie mic și
— preț de cost redus.

Prin urmare, domeniile de utilizare eficientă a microcalculatorului integrat sînt tocmai acelea unde unul din avantajele subliniate mai sus este esențial, adică în cazul:

I. bunurilor tehnice de larg consum
II. unităților de control industrial (controlere)
III. tehnicii de calcul
IV. aparaturii de cercetare, laborator și măsură
V. automatizării în sectoare grele (minier, subacvatic, aviație, nuclear).

Bunurile de larg consum, de exemplu, nu reprezintă numai un subdomeniu de utilizare eficientă a microcalculatorului integrat ci și un loc unde automatizarea prin tehnica de calcul nu poate pătrunde fără existența unui dispozitiv care să integreze toate funcțiile unui microcalculator într-un singur circuit fizic. Încercarea, care s-a făcut, de a implementa unitatea de control și comandă a unei mașini de cusut (ce trebuie în ansamblu să coste cel mult 3000—4000 lei) cu un sistem microprocesor, fie el bazat și pe cel mai bine cotat microprocesor de uz general de 8 biți (Z80), conduce la un volum și la un preț care face inutil orice demers.

Între categoriile de bunuri de larg consum susceptibile de a beneficia de microcalculatorul integrat, sub raportul creșterii performanțelor, a gradului de automatizare și a reducerii componentelor mecanice, enumerăm:

— mașini de spălat automate,

— mașini de gătit cu microunde,
— lăzi frigorifice și instalații de climatizare,
— mașini de cusut,
— automobile (aprindere, carburație, supraveghere),
— aparate cine și fotografice,
— aparate radio, TV și instalații muzicale,
— cîntare,
— jocuri electronice ș.a.

În ceea ce privește unitățile de control industrial aplicațiile în care s-a impus și se va impune microcalculatorul integrat sînt:

— comenzi numerice la mașini-unelte,
— unități de control la roboții industriali,
— reglaj numeric direct în diferite procese,
— comandă și reglaj în electronica de putere,
— supraveghere și alarmă ș.a.

În ceea ce privește tehnica de calcul, microcalculatorul va arenda sfera unităților de control a perifericelor precum:

— trasoare digitale,
— digitizoare,
— imprimante,
— benzi magnetice,
— case de marcat (culegere de date),
— cititoare optice de caractere,

și poate sfera elementelor de procesare paralelă din calculatoarele vectoriale și matriciale.

Motivul pentru care cred că microcalculatorul integrat va ocupa locurile elementelor de procesare paralelă este acela că în cazul calculatoarelor vectoriale sau matriciale puterea și viteza acestora depind de numărul de elemente puse să lucreze în paralel; ori pentru a lega zeci sau chiar sute de asemenea elemente într-o rețea, aceste elemente trebuie reduse fizic la minimum. Nimeni nu-și poate permite un astfel de calculator matricial avînd ca element de procesare paralelă un sistem microprocesor bazat pe INTEL 8080.

Se știe că orice sistem microprocesor minimal cu microprocesorul I8080 trebuie să aibă pe lîngă acesta și circuitul de ceas (8224), circuitul de formare și amplificare a magistralelor (8228) plus memorie citește/scrie și memorie fixă, adică în jur de 8—12 circuite integrate [7]. Ori între un astfel de sistem microprocesor și un microcalculator integrat ca element de procesare paralelă evident că se va impune microcalculatorul integrat. [14].

Avînd în vedere dezvoltarea foarte susținută a microelectronicii, chiar dacă într-o primă fază de experimentare microcalculatorul integrat de uz general va fi folosit ca element de procesare paralelă,

ulterior se va ajunge la microcalculatorul integrat specializat ca element de procesare paralelă. Deocamdată sînt cîteva țări și firme în lume care își pot proiecta și realiza asemenea circuite funcționale (transputere). Cu această precizare și rezervă putem însuma sferei de aplicații din domeniul tehnicii de calcul a microcalculatorului integrat și procesarea paralelă. În prezentul volum ne referim la microcalculatorul integrat de uz general.

Aparatura de cercetare, laborator și măsură impunînd aceleași condiții de preț și volum redus se va dezvolta, perfecționa și automatiza în continuare pe baza microcalculatorului integrat. Între acestea intră:

— aparatură medicală, dozimetre,
— spectrometre de masă, filtre interferențiale,
— generatoare de frecvență și cuvinte, frecventmetre, tahometre etc.

În sectoarele grele sau speciale se mizează în primul rînd pe fiabilitatea, volumul mic, consumul redus al microcalculatorului integrat și numai după aceea pe prețul redus, ca deosebire față de aplicațiile din sfera bunurilor de larg consum. Automatizarea echipamentului de control din: aviație, rachete și sateliți, nave și submarine, centrale nucleare, va apela după cazul aplicației concrete și la microcalculatorul integrat.

1.3. MICROCALCULATOARE INTEGRATE

Între microcalculatoarele integrate mai cunoscute se enumeră și INTEL 8048/8035, MOTOROLA 6801 și 6805 și familia ZILOG Z8.

1.3.1. INTEL 8048/8035

Intel 8048 a fost primul microcalculator integrat; acest dispozitiv conține într-o capsulă de 40 de pini următoarele:

— o unitate centrală de 8 biți,
— o memorie RAM de 64 octeți,
— o memorie ROM de 1 koctet,
— un număr de 27 de linii de I/E,
— un oscilator pilot și
— un circuit de ceas de 8 biți.

În figura 1.1 care reprezintă diagrama bloc a circuitului 8048 sînt evidențiate toate resursele hard înglobate. Conexiunile exterioare ale microcalculatorului 8048 sînt prezentate în figura 1.2.

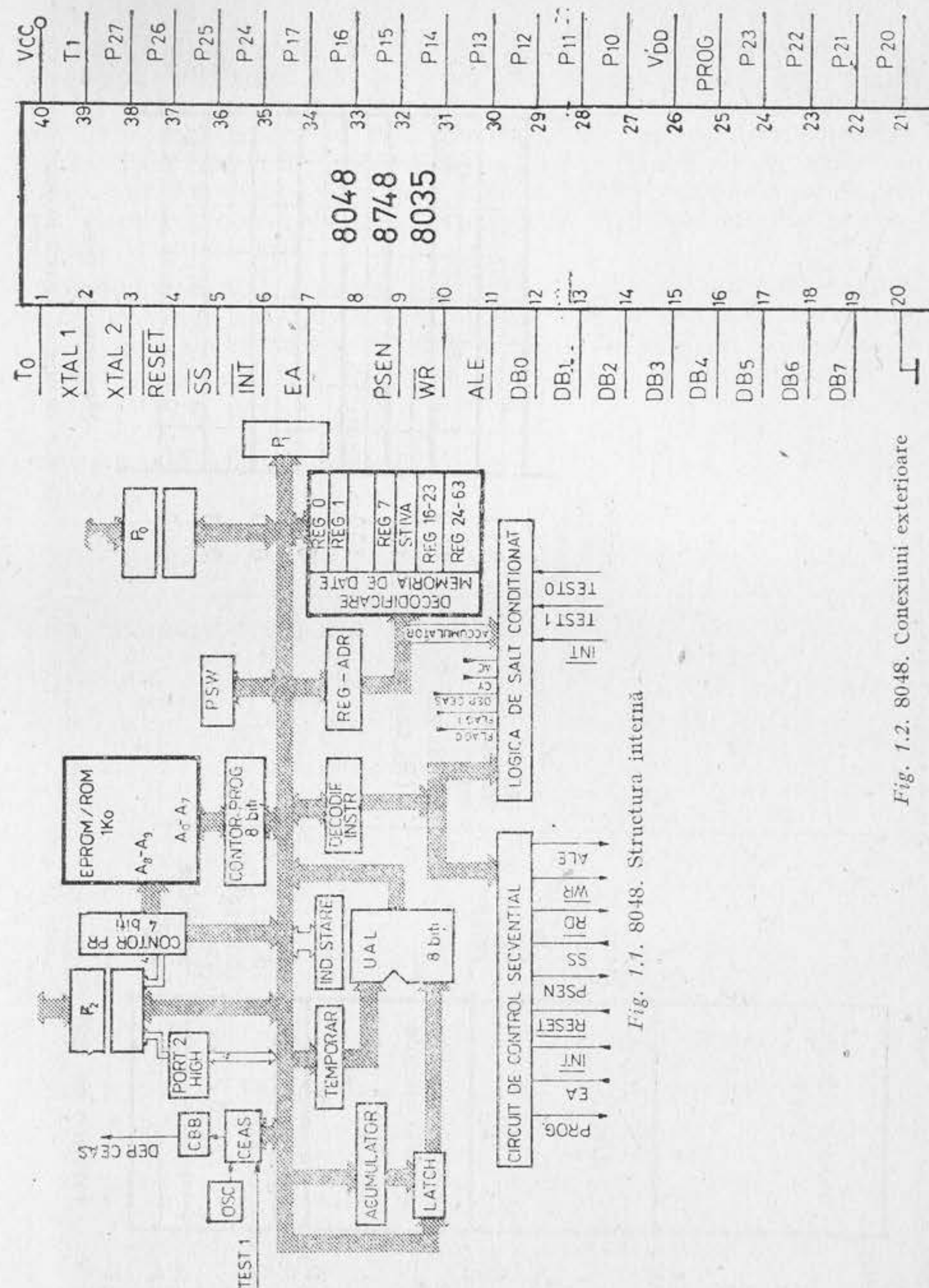


Fig. 1.1. 8048. Structura internă

Fig. 1.2. 8048. Conexiunii exterioare

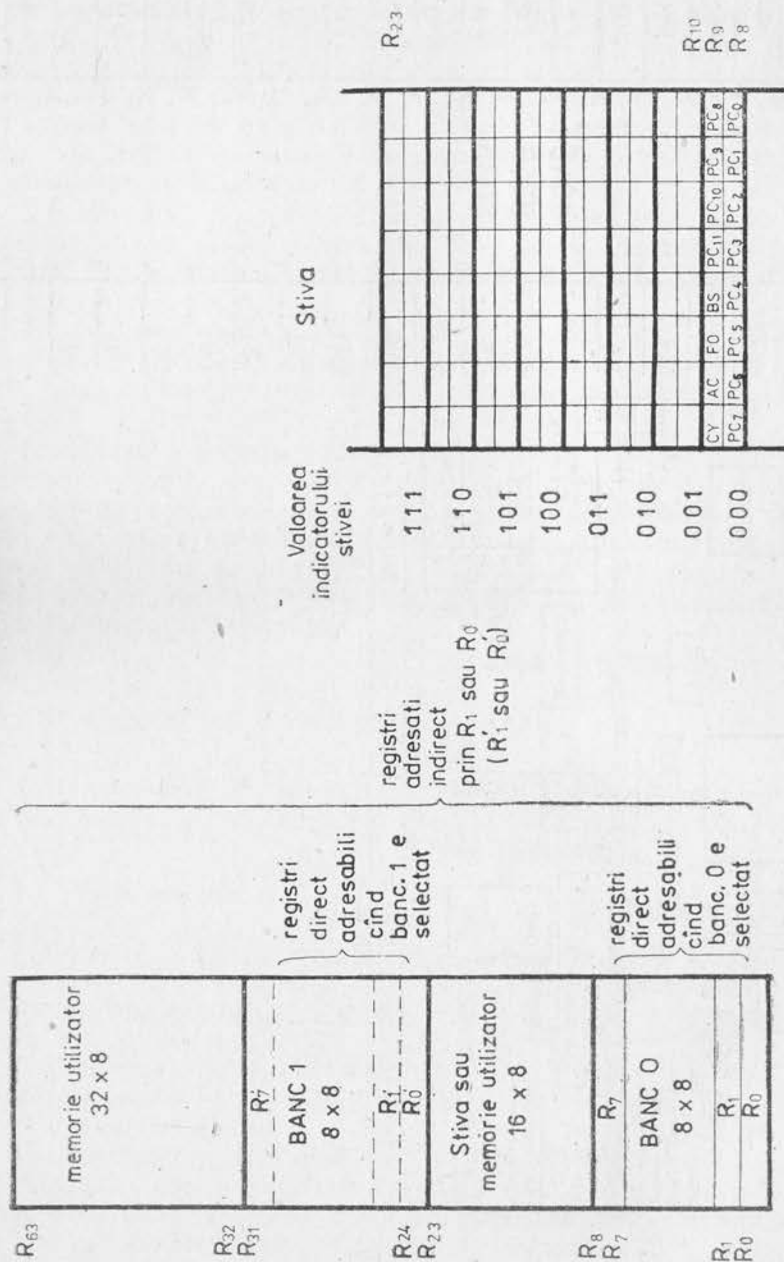


Fig. 1.3. Memoria de date

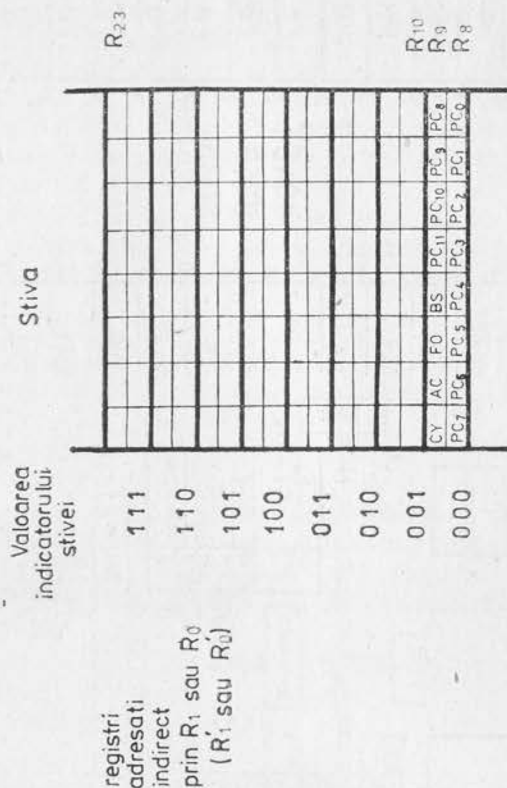


Fig. 1.4. Aranjarea datelor în stivă

Organizarea memoriei de date de 64 de octeţi implementată este prezentată în figura 1.3., iar aranjarea datelor în stivă este redată în figura 1.4. Microcalculatorul integrat 8048 are un ciclu de timp de 2,5 microsecunde sau de 5 microsecunde; repertoriul de instrucţiuni cuprinde 96 de instrucţiuni, avînd fiecare unul sau două cicluri. Setul de instrucţiuni este tributar într-o oarecare măsură primului microprocesor de uz general I 8080, cum este de altfel şi concepţia hardware-ului.

Instrucţiunile sînt grupate astfel: 1) operaţii cu acumulatorul, 2) transferuri de date, 3) salturi, chemări şi reveniri din subrutine, 4) operaţii de intrare/ieşire şi 5) operaţii de control cu registre şi indicatori. În tabelele 1.1. a), b), c), d) şi e) se prezintă setul de instrucţiuni grupat pe tipurile de operaţii.

Tabelul 1.1. SETUL DE INSTRUCŢIUNI

a) OPERAŢII CU ACUMULATORUL

Nr.	Mnemonica	Cod	Octeţi	Cicluri	Descriere
1	ADD A,Rr	0110.1rrr	1	1	(A) ← (A) + (Rr)
2	ADD A,@Rr	0110.000r	1	1	(A) ← (A) + ((Rr))
3	ADD A, #data	0000.0011 data	2	2	(A) ← (A) + data
4	ADDC A,Rr	0111.1rrr	1	1	(A) ← (A) + (Rr) + (C), rrr=0—7
5	ADDC A,@Rr	0111.000r	1	1	(A) ← (A) + ((Rr)) + (C), r=0,1
6	ADDC A,#data	0001.0011 data	2	2	(A) ← (A) + data + (C)
7	ANL A,Rr	0111.1rrr	1	1	(A) ← (A) AND (Rr)
8	ANL A,@Rr	0101.000r	1	1	(A) ← (A) AND ((Rr))
9	ANL A, #data	0101.0011 data	2	2	(A) ← (A) AND data
10	ORL A,R	0100.1rrr	1	1	(A) ← (A) OR (Rr)
11	ORL A,@R	1101.000r	1	1	(A) ← (A) OR ((Rr))
12	ORL A,#data	0100.0011 data	2	2	(A) ← (A) OR data
13	XRL A,R	1101.1rrr	1	1	(A) ← (A) XOR (Rr)
14	XRL A,@R	1101.000r	1	1	(A) ← (A) XOR ((Rr))
15	XRL A,#data	1101.0011 data	2	2	(A) ← (A) XOR data
16	INC A	0001.0111	1	1	(A) ← (A) + 1
17	DEC A	0000.0111	1	1	(A) ← (A) — 1
18	CLR A	0010.0111	1	1	(A) ← 0
19	CPL A	0011.0111	1	1	(A) ← NOT A
20	DA A	0101.0111	1	1	ajustare zecimală
21	SWAP A	0100.0111	1	1	interschimbare semiocteţi

(continuare **Tabelul 1.1.**)

Nr.	Mnemonica	Cod	Octeţi	Cicluri	Descriere
22	RL A	1110.0111	1	1	(A0) _ (A7), (An+1) _ (An), n=0−6
23	RLC A	1111.0111	1	1	(An+1) _ (An), (A0) _ (C), (C) _ A7
24	RR A	0111.0111	1	1	(An) _ (An+1), (A7) _ (A0)
25	RRC A	0110.0111	1	1	(An) _ (An+1), (A7) _ (C), (C) _ (A6)

b) TRANSFERURI DE DATE

Nr.	Mnemonica	Cod	Octeţi	Cicluri	Descriere
1	MOV A,R	1111.1rrr	1	1	(A) _ (Rr), rrr=0−7
2	MOV A,@R	1111.000r	1	1	(A) _ ((Rr)), r=0,1
3	MOV A,#data	0010.0011 data	2	2	(A) _ data
4	MOV R,A	1010.1rrr	1	1	(Rr) _ (A)
5	MOV R,@A		1	1	((Rr)) _ (A)
6	MOV R,#data	1011.1rrr data	2	2	(Rr) _ data
7	MOV @R,#data	1010.000r data	2	2	((Rr)) _ data
8	MOV A,PSW	1100.0111	1	1	(A) _ (PSW)
9	MOV PSW,A	1101.0111	1	1	(PSW) _ (A)
10	XCM A,R	0010.1rrr	1	1	(A) _ (Rr)
11	XCM A,@R	0010.000r	1	1	(A) _ ((Rr))
12	XCHD A,R	0011.000r	1	1	(A0−3) _ ((Rr0−3))
13	MOVX A,@R	1000.000r	1	2	(A) _ ((Rr))
14	MOVX @R,A	1001.000r	1	2	((Rr)) _ A
15	MOVP A,A	1010.0011	1	2	(PC0−7) _ A, (A) _ ((PC))
16	MOVP3 A,A	1110.0011	1	2	(PCO−7) _ (A), (PC8−10) _ 011, (A) _ ((PC))

(continuare **Tabelul 1.1.**)

c) SALTURI, SUBRUTINE

Nr.	Mnemonica	Cod	Octeţi	Cicluri	Descriere
1	JMP adr	A10A9A8A 70100 adr 0−7	2	2	(PC8−10) _ adr 8−10 (PC0−7) _ adr 0−7 (PC11) _ DBF
2	JMPP A	1011.0011	1	2	(PC) _ ((A))
3	DJNZ R,adr	1110.1rrr adr 0−7	2	2	(R) _ (R) − 1 dacă R = 0 (PC) _ adr
4	JC adr	1111.0110 adr	2	2	(PC) _ adr dacă C = 1 altfel (PC) = (PC) + 2
5	JNC adr	1110.0110 adr	2	2	(PC) _ adr dacă C = 0, altfel (PC) = (PC) + 2
6	JZ adr	1100.0110 adr	2	2	(PC) _ adr dacă A = 0 altfel (PC) = (PC) + 2
7	JNZ adr	1001.0110 adr	2	2	(PC) _ adr dacă A = 0 altfel (PC) = (PC) + 2
8	JT0 adr	0011.0110 adr	2	2	(PC) _ adr dacă T0 = 1 altfel (PC) = (PC) + 2
9	JNT0 adr	0010.0110 adr	2	2	(PC) _ adr dacă T0 = 0 altfel (PC) = (PC) + 2
10	JT1 adr	0101.0110 adr	2	2	(PC) _ adr dacă T1 = 1 altfel (PC) = (PC) + 2
11	JNT1 adr	1001.0110 adr	2	2	(PC) _ adr dacă T1 = 0 altfel (PC) = (PC) + 2
12	JF0 adr	1011.0110 adr	2	2	(PC) _ adr dacă F0 = 1 altfel (PC) = (PC) + 2
13	JF1 adr	0111.0110 adr	2	2	(PC) _ adr dacă F1 = 1 altfel (PC) = (PC) + 2
14	JTF adr	0001.0110 adr	2	2	(PC) _ adr dacă TF = 1 altfel (PC) = (PC) + 2
15	JNI adr	1000.0110 adr	2	2	(PC) _ adr dacă I = 0 altfel (PC) = (PC) + 2
16	JBb adr	B2B1B01. 0010 adr 0−7	2	2	(PC0−7) _ adr dacă Bb = 1 altfel (PC) = (PC) + 2
17	CALL adr	A10A9A8101 00 adr 0−7	2	2	((SP)) _ (PC), (SPW 4−7) (SP) _ (SP) + 1 (PC 8−10) _ adr 8−10 (PC 0−7) _ adr 0−7 PC 11 _ DBF
18	RET	1000.0011	1	2	(SP) _ (SP) − 1 (PC) _ ((SP))
19	RETR	1001.0011	1	2	(SP) _ (SP) − 1 (PC) _ ((SP)) (PSW 4−7) _ ((SP))

(continuare **Tabelul 1.1.**)

d) OPERAȚII DE INTRĂRI/IEȘIRI

Nr.	Mnemonica	Cod	Octeți	Cicluri	Descriere
1	IN A,P	0000.10pp	1	2	(A) _ (Pp) p=1,2
2	OUTL P,A	0011.10pp	1	2	(Pp) _ (A)
3	ANL P #data	1001,10pp data	2	2	(Pp) _ (Pp) AND data
4	ORL P,#data	1000.10pp data	2	2	(Pp) _ (Pp) OR data
5	INS A, BUS	0000.1000	1	2	(A) _ (BUS) strobat RI
6	OUTL BUS,A	0000.0010	1	2	(BUS) _ (A)
7	ANL BUS,#data	1001.1000 data	1	2	(BUS) _ (BUS) AND data
8	ORL BUS,#data	1000.1000 data	2	2	(BUS) _ (BUS) OR data
9	MOVD A,P	0000.11pp	1	2	(A 0—3) _ (Pp) , (A 4—7) _ 0 p = 4,7
10	MOVD P,A	0011.11pp	1	2	(Pp) _ (A 0—3)
11	ANLD P,A	1001.11pp	1	2	(Pp) _ (Pp) AND (A 0—3) p = 4,7
12	ORLD P,A	1000.11pp	1	2	(Pp) _ (Pp) OR (A 0—3) p = 4,7
13	MOV A,T	0100.0010	1	1	(A) _ (T)
14	MOV T,A	0110.0010	1	1	(T) _ (A)
15	STRT T	0101.0101	1	1	pornirea circuitului de ceas
16	STRT CNT	0100.0101	1	1	activarea numărătorului de evenimente externe
17	STOP TCNT	0110.0101	1	1	oprirea circuitului de ceas
18	EN TCNTI	0010.0101	1	1	validarea INT de la ceas
19	DIS TCNTI	0011.0101	1	1	dezactivarea INT de la ceas
20	ENTO CLK	0111.0101	1	1	pin TO devine ieșire de TACT

e) OPERAȚII DE CONTROL CU REGISTRE ȘI INDICATORI

Nr.	Mnemonica	Cod	Octeți	Cicluri	Descriere
1	INC R	0001.1rrr	1	1	(R) _ (R) + 1, rrr = 0—7
2	INC @R	0001.000r	1	1	((R)) _ ((R)) + 1, r = 0,1
3	DEC R	1100.1rrr	1	1	(Rr) _ (Rr) — 1
4	CLR C	1001.0111	1	1	C _ 0
5	CPL C	1010.0111	1	1	C _ NOT C
6	CLR F0	1010.0101	1	1	F1 _ 0
7	CPL F0	1001.0101	1	1	F0 _ NOT F0
8	CLR F1	1000.0101	1	1	F0 _ 0

(continuare **Tabelul 1.1.**)

Nr.	Mnemonica	Cod	Octeți	Ciclari	Descriere
9	CPL F1	1011.0101	1	1	F1 _ NOT F1
10	EN 1	0000.0101	1	1	activarea INT externe
11	DIS 1	0001.0101	1	1	dezactivarea INT externe
12	SEL RB0	1100.0101	1	1	BS _ 0
13	SEL RB1	1101.0101	1	1	BS _ 1
14	SEL MB0	1110.0101	1	1	(PC 11) _ 0
15	SEL MB1	1111.0101	1	1	(PC 11) _ 1
16	NOP	0000.0000	1	1	execuția continuă cu următoarea instrucțiune

Pentru a reduce timpul de trecere de la faza de cercetare — dezvoltare a unui produs la introducerea lui în fabricație, firma INTEL a creat două versiuni compatibile pînă la nivel de pin: versiunea cu ROM inter ı, 8048, și versiunea cu memorie internă reprogramabilă (EPROM), 8748. Circuitul 8748, destinat fazei de punere la punct a proiectelor, poate fi înscris cu un anumit conținut și remodificat de mai multe ori pe parcursul acestei faze, după care va fi înlocuit de versiunea 8048 în producția de serie.

În jurul microcalculatorului 8048 s-au dezvoltat și o serie de circuite auxiliare care sînt prezentate sintetic în tabelul nr. 1.2.. Memoria program și de date a microcalculatorului poate fi extinsă fie utilizînd memorii standard fie folosind memoriile 8355 și 8155 care includ și linii de I/E programabile și circuite de ceas. Liniile de I/E pot fi extinse folosind circuitul 8243. Este de remarcat că 8048 poate fi extins și cu circuite din familia „vechiului" și cunoscutului microprocesor 8080.

Circuitul 8035 este o altă versiune a microcalculatorului 8048 fără memorie program internă dar la care se poate atașa o memorie externă de dimensiunile cerute de aplicația dată.

Tabelul 1.2. FAMILIA MICROCALCULATORULUI 8048

microcalculatoare	8048 8748 8035	memorie program fixă ROM memorie progr. EPROM memorie program externă
memorii și circuite	8355 8755 8155	2ko ROM, 16 linii de I/E 2ko EPROM, 16 limii de I/E 256 octeți RAM, 22 linii de I/E
circuite de I/E	8243	16 linii de I/E

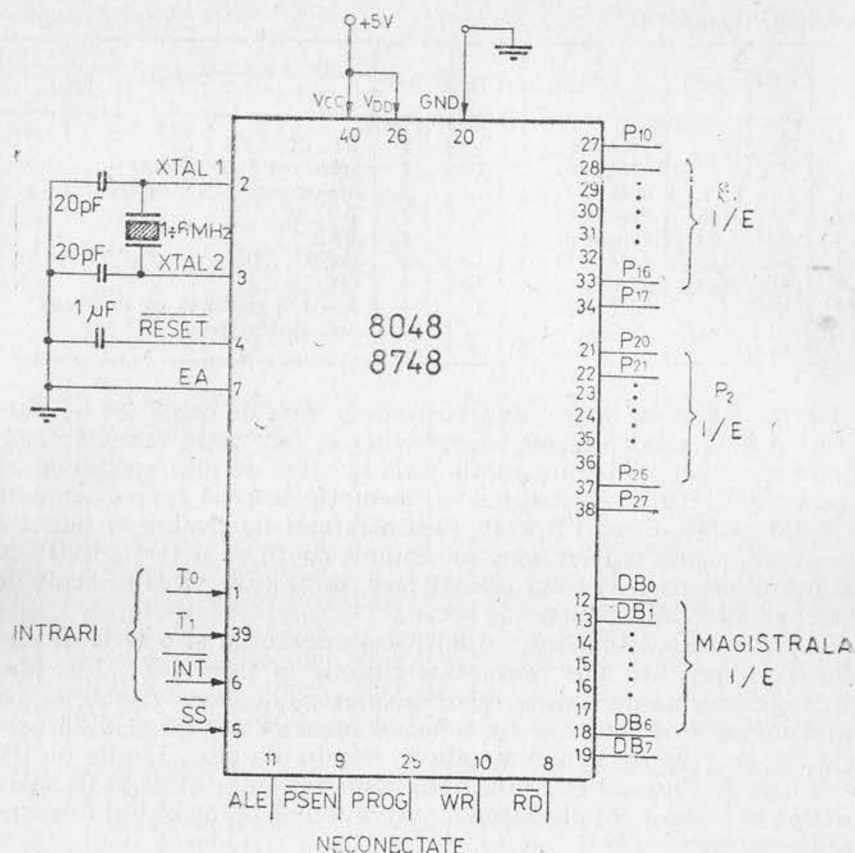


Fig. 1.5. 8048 în regim de sine stătător

În figura 1.5. se prezintă microcalculatorul 8048 în regim de sine stătător.

1.3.2. MOTOROLA 6801

MC6801 este un microcalculator integrat de 8 biţi care s-a dezvoltat din familia microprocesorului de uz general MC6800. Instrucţiunile microcalculatorului integrat MC6801 sînt compatibile cu instrucţiunile microprocesorului mamă. Timpul de execuţie a fost redus şi cîteva noi instrucţiuni au fost adăugate; între acestea şi

instrucţiunea de înmulţire fără semn. MC6801 poate funcţiona fie ca microcalculator de sine stătător fie ca microcalculator de uz general ce poate utiliza o memorie externă de cel mult 64 ko.

MC6801 include pe lîngă microprocesorul propriu-zis:

— o memorie fixă de 2ko,
— o memorie citeşte/scrie de 128 octeţi,
— un număr de 29 linii de I/E,
— trei circuite de ceas programabile de 16 biţi fiecare şi
— un oscilator pilot.

Schema bloc de principiu din care se văd resursele circuitului este prezentată în figura 1.6. Porturile P3 şi P4 sînt multifuncţionale şi ca atare pot fi configurate pentru funcţionarea microcalculatorului în regim de sine stătător, sau în regim extins nemultiplexat, sau în regim extins multiplexat. În tabelul nr. 1.3. sînt prezen-

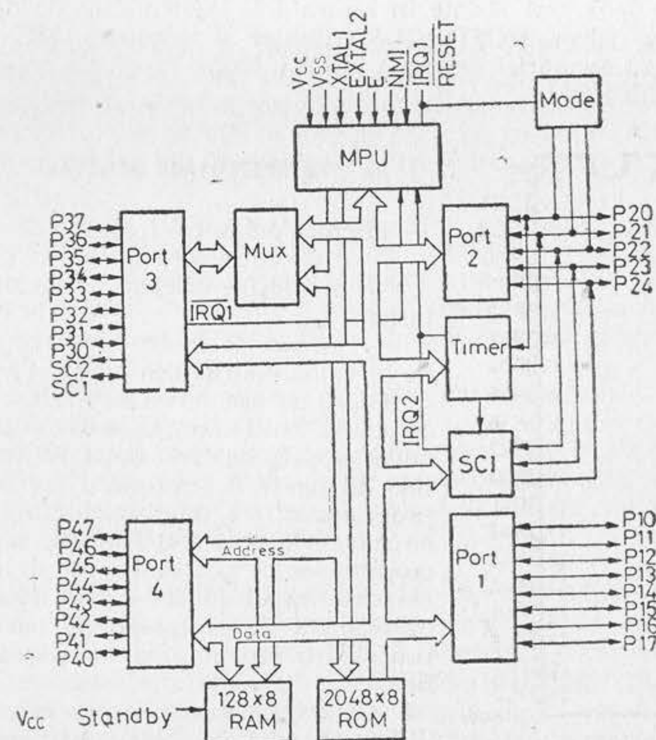


Fig. 1.6. MC6801. Structura internă

Tabelul 1.3. MODUL DE CONFIGURARE A PORTURILOR P3 și P4

P3	P37	P36	P35	P34	P33	P32	P31	P30	SC2	SC1
*	I/E7	I/E6	I/E5	I/E4	I/E3	I/E2	I/E1	I/E0	OS3	IS3
**	D7	D6	D5	D4	D3	D2	D1	D0	R/W	IOS
***	A/D7	A/D6	A/D5	A/D4	A/D3	A/D2	A/D1	A/D0	R/W	AS
P4										
*	I/E7	I/E6	I/E5	I/E4	I/E3	I/E2	I/E1	I/E0		
**	A7	A6	A5	A4	A3	A2	A1	A0		
***	A15	A14	A13	A12	A11	A10	A9	A8		

unde: * microcalculator de sine stătător
** regim extins nemultiplexat
*** regim extins multiplexat

tate funcțiile acestor porturi la nivel de bit în cele trei regimuri posibile de funcționare. Conexiunile exterioare ale microcalculatorului MC 6801 sînt redate în figura 1.7. Pentru faza de dezvoltare de produse firma MOTOROLA produce și versiunea MC 68701 ce are în locul memoriei program interne fixe o memorie programabilă electric, EEPROM.

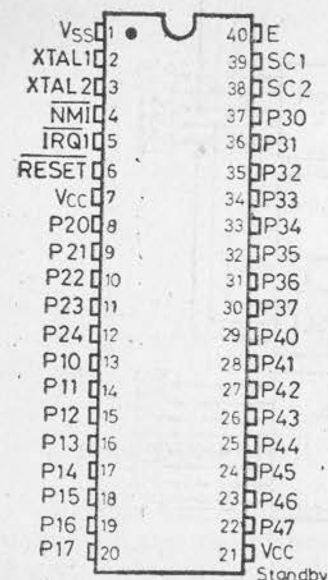


Fig. 1.7. MC6801. Conexiuni exterioare

1.3.3. Z8. PREZENTARE GENERALĂ

Microcalculatorul integrat Z8 oferă:

— o execuție rapidă,

— o folosire eficientă a memoriei interne,

— un set de instrucțiuni ce pune bine în valoare structura hardware.

Oferă de asemenea facilitatea „croirii" structurii hardware (în anumite limite) pe nevolle aplicației sub controlul programului. Z8 poate fi configurat sub controlul programului ca microcalculator integrat cu memorie fixă 2(4) Kocteți, sau ca microprocesor ce poate dispune de o memorie externă de [128 — 2(4)] Kocteți sau ca element de procesare dintr-o rețea de procesoare legate prin magistrale Z — Bus.

Caracteristicile microprocesorului Z8 îl fac, în mod deosebit, util pentru aplicațiile de control în timp real.

Între caracteristicile care-l recomandă pentru acest gen de aplicații trebuie menționate:

— timpul de execuție mediu mic 1,5—2,5 μsec la un cuarț de 8MHZ,

— răspuns rapid la întreruperi [11 μs, timp în care este inclusă și salvarea automată a indicatorilor de stare],

— tratarea a 6 întreruperi vectorizate cu posibilități de mascare și de stabilire a priorității dorite,

— prezența în structura microcalculatorului a circuitului de intrare/ieșire serie și a două circuite de timp cu o gamă diversă de utilizări posibile.

— un set de instrucțiuni (43 de tipuri, 231 de coduri) bine ales ce utilizează foarte eficient resursele integrate.

1.3.4. FAMILIA ZILOG — Z8

Familia Z8 cuprinde o variantă standard Z8611 și mai multe versiuni de dezvoltare. Din punct de vedere al structurii interne diferențele sînt minime și privesc numai capacitatea de memorie integrată sau nu pe așchia de SI; în ceea ce privește setul de instrucțiuni nu există nici o diferență între microcalculatoarele familiei.

Z8611 — varianta standard. Are o memorie internă ROM de 4 ko, înscrisă de producător la cerere în momentul realizării circuitului. Este încapsulată în pastila de 40 de pini. Toate celelalte versiuni se bazează pe aceasta, diferențele fiind specificate cînd apar.

Z8612 — versiunea de dezvoltare cu interfața de memorie. Memoria program internă nu mai este integrată, în locul ei apare o interfață de legătură la pinii externi. Astfel utilizatorul are în exterior memoria EPROM sau RAM dorită. Din cauza conexiunilor exterioare necesitate de conectarea memoriei program această versiune este încapsulată în pastile de 64 de pini.

Z8613 — versiunea de prototip cu interfața de EPROM (Prototyping Device with EPROM Interface). Această versiune este compatibilă pin la pin cu varianta standard Z8611. Memoria program nu este integrată și are o interfață cu un EPROM tip INTEL 2732 scoasă într-un soclu montat direct pe spatele capsulei. În acest fel se asigură compatibilitatea și interschimbabilitatea cu varianta cu programul „ars", care e destinată producției de serie. Soclul din spa-

tele capsulei, de 24 pini, are 12 linii de adrese, 8 linii de date, celelalte linii fiind +5V, GND, selecție capsulă (CS) și validare ieșire (DE). Pornind de la această versiune se poate dezvolta simultan produsul cu pregătirea fabricației — nefiind nevoie de plăci de circuit imprimat diferite. Și în producția de serie mică, unde varianta cu propramul „ars" este prohibitivă ca preț și timp de punere la punct, versiunea Z8613 este atrăgătoare.

Z8671 — versiunea cu interpretor BASIC integrat. Această versiune este un microcalculator cu interpretor/depanator BASIC încorporat care se interfațează cu utilizatorul prin legătura serie. Folosind în acest scop circuitele de ceas și de intrare/ieșire serie proprii.

Interpretorul BASIC poate adresa direct regiștri interni și memoria externă. Poate examina și modifica ușor orice locație de memorie sau registru de intrare/ieșire. Interpretorul BASIC poate chema subrutine scrise în limbaj mașină necesare aplicațiilor critice din punct de vedere al timpului. Depanatorul este interactiv datorită editorului pe care-l conține și care lucrează în modul „imediat". Astfel Z8671 oferă o combinație de resurse hardware și software care-l fac atractiv în multe aplicații industriale.

Z8671 este încapsulat în aceeași pastilă cu varianta standard Z8611.

PROGRAMUL DE FABRICAȚIE AL MICROCALCULATOARELOR Z8 AL FIRMEI MIKROELEKTRONIK ERFURT

Acest program de fabricație cuprinde și variantele standard și de dezvoltare ale firmei ZILOG deja prezentate dar mai cuprinde încă și alte versiuni funcție de capacitatea de memorie, de opțiunea de consum redus și de frecvența maximă de lucru. În continuare prezentăm pe scurt acest program:

UB 8810 D — microcalculator integrat Z8 cu memorie ROM internă (40 pini)

UB — 8811 D — același cu UB 8810 D dar necesitînd tact extern, avînd în schimb opțiunea de consum redus (40 pini)

UB 8820 M — microcalculator integrat Z8, versiunea de dezvoltare cu interfața pentru memorie externă de 2 ko (64 pini)

UB 8821 M — același cu UB 8820 M cu opțiunea de consum redus

UB 8830 D — microcalculator integrat Z8 cu încărcător și interpretor BASIC integrat (40 pini)

UB 8831 D — același cu UB 8830 D cu opțiunea de consum redus

UB 8840 M — microcalculator integrat Z8, versiunea de dezvoltare cu interfața pentru memorie externă de 4ko (64 pini)

UB 8841 M — același cu UB 8840 M cu opțiunea de consum redus

UB 8860 D și UB 8861 D — microcalculator integrat Z8 fără memorie internă ROM și fără interfața cu memorie externă, încapsulată în varianta cu 40 de pini. Caracteristicile acestei versiuni sînt următoarele:

— la un nivel între 7,35V — 8 V pe intrarea de RESET se face un salt la memoria program externă prin intermediul porturilor PO și P1,

— 43 de tipuri de instrucțiuni,

— 124 de registre de uz general, 4 registre de I/E, 16 registre de control și configurare,

— 32 de linii de intrare/ieșire,

— un circuit de intrare/ieșire serie,

— 2 circuite de ceas (fiecare ceas avînd un divizor programabil de 8 biți și un predivizor de 6 biți),

— 6 întreruperi vectorizate, mascabile și aranjabile în șir de priorități,

— timp de execuție mediu a instrucțiunilor de 2,2 μs.

1.3.5. CARACTERISTICILE MICROCALCULATOARELOR UB 88XX

VALORILE LIMITĂ

Tensiune de alimentare	Ucc −0,5V ÷ 7V
Tensiune de intrare	Ui −0,5V ÷ 7V
Tensiune de ieșire	Uo −0,5V ÷ 7V
Domeniu de temperatură în funcționare	Tf 0° ÷ 70°C
Domeniu de temperatură de stocare	Tst −55°C ÷ 125°C

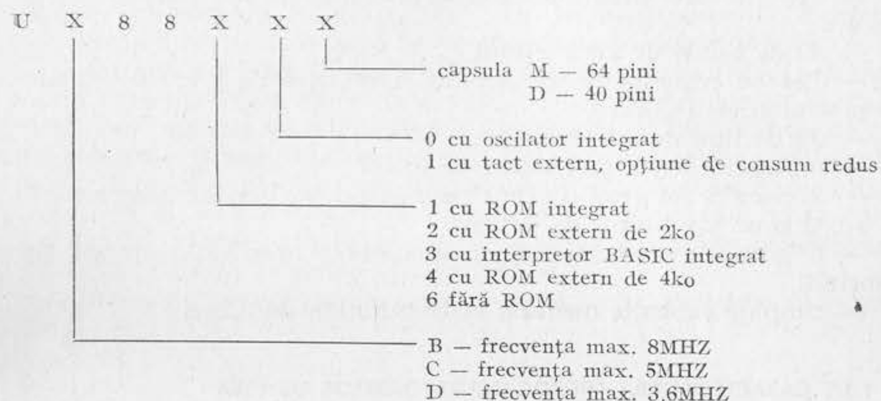
CARACTERISTICILE STATICE (Tf = 0 — 70°C)

Tensiune de alimentare	Ucc = 4,75V ÷ 5,25V Umm = U_{cc} — 0,6 ÷ Ucc
Tensiune de alimentare în consum redus	Ucc = 0 ÷ 4,75V Umm = 3V ÷ 5,25V
Tensiune de intrare	UIL = −0,3V ÷ 0,8V UIH = 2V ÷ Ucc
Nivelul tactului	UTIL = −0,3 V ÷ 0,8V UTIH = 3,8V ÷ Ucc

CARACTERISTICI DINAMICE

Frecvența de intrare	$f_c = 1 \div 8$ MHZ
Timp de front pozitiv și negativ	t_f = max 25ns
Lățimea tactului	t_i = minim 37ns

ÎNTOCMIREA SIMBOLULUI DE COMANDĂ



Întrucît microcalculatoarele integrate Z8, indiferent de sursă și de versiune sînt identice din punct de vedere funcțional, în prezentarea care urmează se fac referiri la acest microcalculator sub denumirea generică de Z8. Luînd în considerare diferențele care apar în această familie, membrii acesteia se pot grupa în: — versiunile cu memorie înglobată (40 de pini) care vor fi menționate, pe scurt, în continuare sub forma Z8 sau Z8/40,

— versiunile de dezvoltare cu interfață pentru memorie externă (64 de pini), care vor fi specificate pe scurt sub forma Z8/64.

1.4. MICROCONTROLERE

O categorie aparte de microcalculatoare integrate sînt microcontrolerele. Nu există diferență netă între cele două categorii. Aceste microcontrolere, prin setul de instrucțiuni mai redus, sînt mai intim legate de aplicațiile de control, urmărire și automatizare industrială. Microcalculatoarele integrate de uz general vin dinspre

minicalculatoare spre aplicațiile industriale. Bineînțeles că și microcontrolerele actuale au încorporate pe lîngă procesorul (controlerul) propriu-zis: memorie citește/scrie, memorie fixă, circuite de intrare/ieșire, circuite de ceas și oscilatorul pilot — pentru a fi utilizabile în anumite aplicații direct, fără multe circuite în jur. De aceea microcontrolerele sînt foarte atractive în anumite aplicații industriale, precis delimitate, cu caracter final. În orice caz fiecare aplicație trebuie cîntărită bine și evaluată corect pentru a o dezvolta fie în jurul unui microcontroler, fie în jurul unui microcalculator integrat. Mai departe nu vom utiliza sintagma de microcontrolere integrate pentru că nu este uzuală, deși în felul acesta s-ar face o deosebire față de primele controlere apărute și ar fi în concordanță cu sintagma similară de microcalculator integrat. În continuare vom prezenta o familie de microcontrolere produse de firma MICROCHIP TECHNOLOGY.

1.4.1. FAMILIA PIC 16C5X

Microcontrolerul PIC 16C5X, ca orice microcalculator integrat conține pe lîngă procesorul propriu-zis și memoria fixă, EPROM, memoria citește/scrie, linii de intrare/ieșire, circuit de ceas și oscilatorul pilot. Folosind o tehnologie CMOS EPROM rapidă, firma oferă o familie de microcalculatoare extrem de flexibile, cu cost de cercetare redus și timp de dezvoltare scurt. Adică tocmai dezideratele oricărui proiectant de sisteme logice. De asemenea consumul extrem de redus și domeniul larg al tensiunilor de alimentare, recomandă această familie de controlere în domeniul aparaturii portabile, aviației, automobilelor și în domeniul electronicii distribuite. Familia cuprinde o varietate de opțiuni de circuite care se disting prin: dimensiunea EPROM-ului și RAM-ului, numărul de I/E, tipul oscilatorului integrat, domeniul de frecvență și modul de încapsulare. În funcție de cerințele aplicației și ale producției se alege varianta optimă. În tabelul 1.4. se evidențiază capacitatea de memorie, numărul liniilor de intrare/ieșire, tipul oscilatorului și domeniul de frecvență, la cîteva circuite din familie.

CARACTERISTICI GENERALE

— domeniul de frecvență: 0—8 Mhz,
— tensiunea de alimentare: 3,5—6 V,
— consumul: < 2 mA la 4 Mhz, 5 V și < 10 μA în regim standby,

— domeniul termic de funcționare:
var. comercială 0 — +70 C
var. industrială — 40 — +85 C
var. specială —40 — +110 C,
— memoria:
program, fixă (EPROM), 512 (2 ko) × 12 biți
de date, citește/scrie (RAM) 32 (80) × 8 biți,
— setul de instrucțiuni: 33 tipuri de instrucțiuni,
— modurile de adresare: direct, indirect, imediat și relativ,
— liniile de I/E: 12(20) bidirecționale, three state,
— stiva: 2 registre,
— circuitele de ceas: 1 numărător și un predivizor de 8 biți.
— sistemul de urmărire: 1 watchdog s.a.

Tabelul 1.4 FAMILIA PIC 16C5X

	EPROM	RAM	I/E	Alim	Osc.	Frecv.
PIC 16C54RC	512×12	32×8	13	3,5–6V	RC	0–4 Mhz
PIC 16C54HS	512×12	32×8	13	4,5–6V	Cuart	4–8 Mhz
PIC 16C55RC	512×12	32×8	21	3,5–6V	RC	0–4 Mhz
PIC 16C56RC	1k×12	32×8	13	3,5–6V	RC	0–4 Mhz
PIC 16C57RC	2k×12	80×8	21	3,5–6V	RC	0–4 Mhz
PIC 16C57XT	2k×12	80×8	21	3,5–6V	Cuart	0,4–4 Mhz
PIC 16C57HS	2k×12	80×8	21	4,5–6V	Cuart	4–8 Mhz
PIC 16C57LP	2k×12	80×8	21	TBD	Cuart	32khz

ARHITECTURA MICROCONTROLERELOR PIC 16C5X

Structura microcontrolerului este o structură Harvard, bazată pe conceptul de registre de instrucțiuni și registre de date cu magistrale separate. Magistrala de date și memoria de date (RAM) sînt organizate pe 8 biți, în timp ce memoria program (EPROM) este organizată pe 12 biți. Acest concept face ca un set redus de instrucțiuni să fie suficient de puternic; setul de instrucțiuni a fost gîndit pentru procesare la nivel de bit, octet sau registru, la viteze mari și în condiții de suprapunere a ciclului de aducere a instrucțiunii următoare cu ciclul de execuție a instrucțiunii curente. Diagrama bloc a microcontrolerelor PIC este prezentată în fig. 1.8.

SETUL DE REGISTRE.

Setul de registre sau memoria citește/scrie (RAM) este împărțit în 2 grupuri funcționale: registrele operaționale (dedicate) și re-

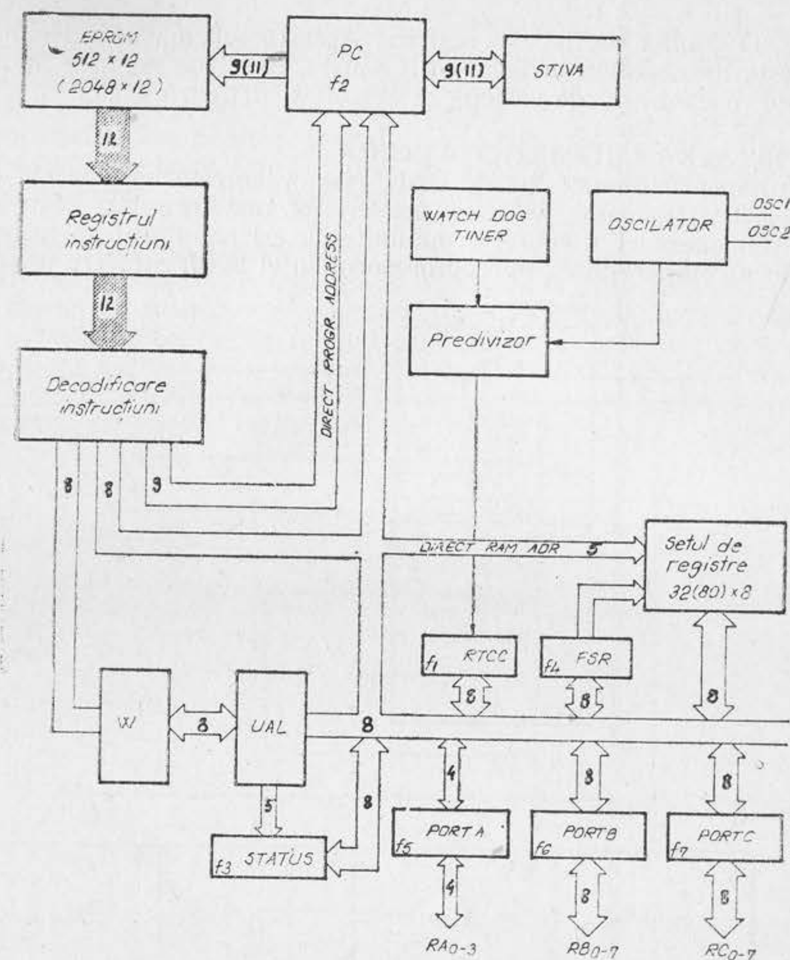


Fig. 1.8. Microcontrolerele PIC 16C5x. Structura internă

gistrele de uz general. În prima categorie intră registrul numărătorului de ceas (RTCC — real time clock counter), contorul program (PC — program counter), registrul de stare, registrele porturilor și registrul de selecție a paginii de registre (FSR — file select register). Magistrala de date de 8 biți conectează setul de registre cu unitatea aritmetico-logică. Primele 32 de registre sînt direct adresabile. Opțiunile cu RAM mai mare (PIC 16C57) adresează registrele ce depășesc numărul de 32 printr-o schemă de selecție a paginii de re-

gistre. O pagină conţine 16 registre. Această schemă de selecţie se foloseşte de registrul dedicat, FSR sau f4. Setul de registre împreună cu mecanismul de selecţie este schiţat în figura 1.9.

UNITATEA ARITMETICO-LOGICĂ.

Unitatea aritmetico-logică, UAL, este organizată pe 8 biţi şi are un registru temporar, W, care are un rol mai deosebit; păstrează unul din operanzi şi reţine în anumite cazuri rezultatul operaţiilor. Făcind o comparaţie cu microprocesorul Intel 8080, care are un sin-

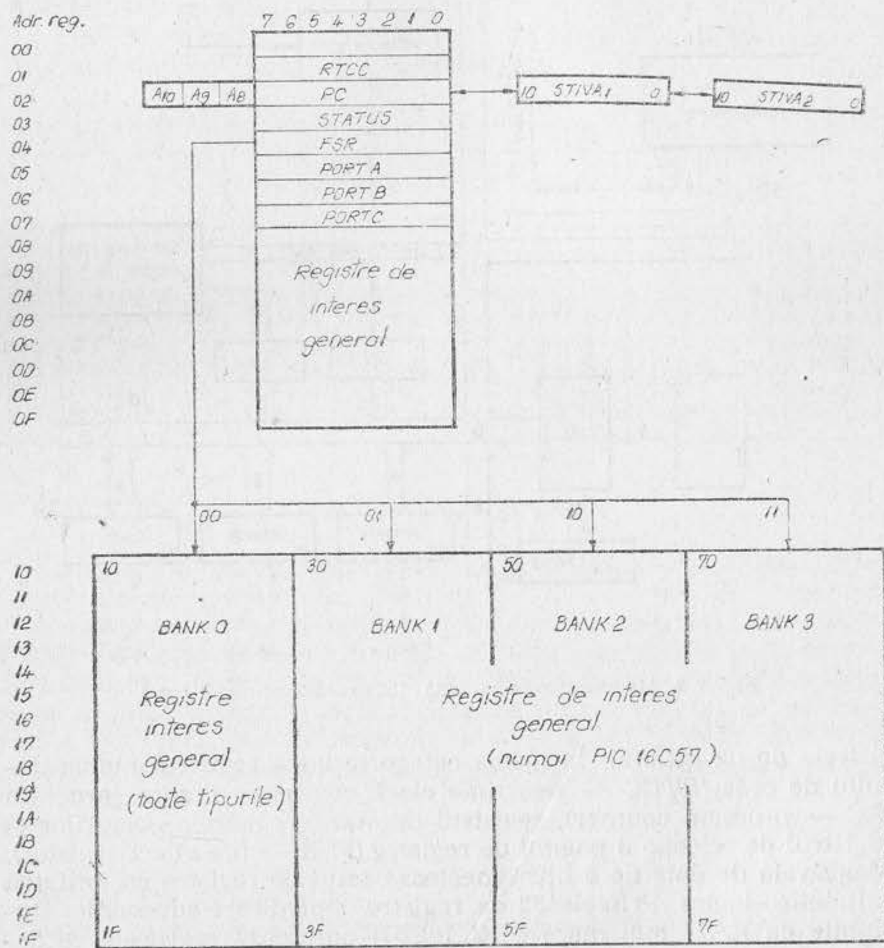


Fig. 1.9. PIC 16CSx. Setul de registre

gur registru dedicat, acumulatorul, pentru stocarea rezultatelor şi situaţia microcalculatorului integrat Z8, unde fiecare registru poate fi şi acumulator, în cazul de faţă sîntem undeva la jumătatea distanţei. UAL execută operaţiile aritmetice şi funcţiile booleene între operanzii din acest registru, W, şi orice alt registru. Registrul W poate fi încărcat imediat prin instrucţiunile „literare" care încarcă date imediat din chiar corpul instrucţiunii.

MEMORIA PROGRAM.

Memoria program conţine de la 512 cuvinte pînă la 2048 cuvinte sub formă de memorie reprogramabilă (EPROM), funcţie de opţiunea aleasă. Cuvintele au lungimea de 12 biţi. Variantele de circuit cu opţiuni de memorie program mai mari de 512 cuvinte adresează pagini succesive de 512 cuvinte fiecare. Memoria program împreună cu schema de selecţie este redată în figura 1.10.

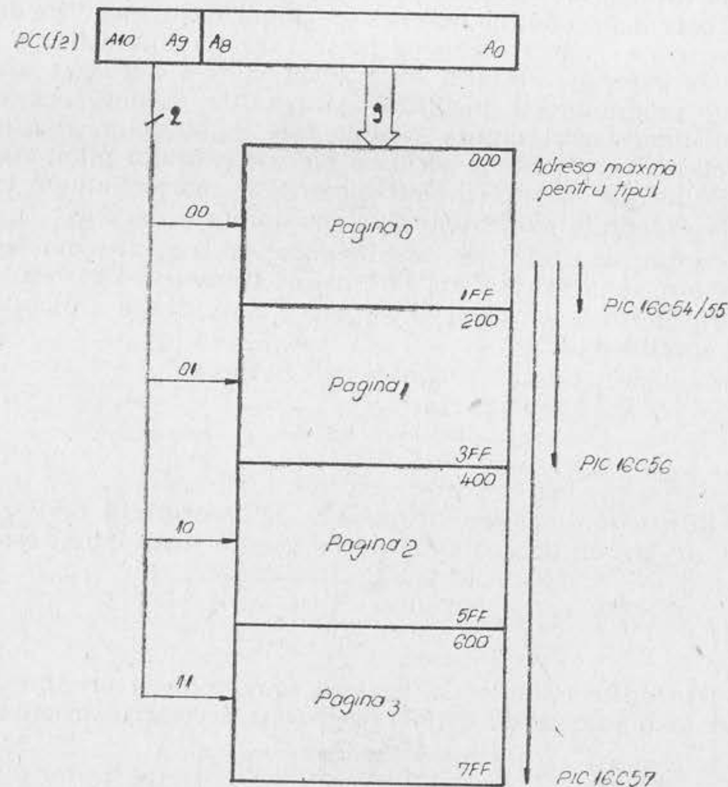


Fig. 1.10. Memoria program şi schema de selecţie

Succesiunea instrucțiunilor este controlată cu ajutorul contorului program PC, care se incrementează automat după fiecare ciclu de aducere a unei noi instrucțiuni. Operațiile prin care se controlează desfășurarea programului, operații care includ modul de adresare direct, indirect și relativ, se realizează prin instrucțiuni de test pe bit, de salt, de chemare subrutină sau de încărcare imediată a contorului program. Legat de contorul program există două registre de stivă care permit un nivel de imbricare (chemarea unei a doua subrutine dintr-o primă subrutină).

SETUL DE INSTRUCȚIUNI.

Fiecare instrucțiune este un cuvînt de 12 biți divizat, în general, în 3 cîmpuri: codul de operație și 2 operanzi. Instrucțiunile se pot împărți formal în: instrucțiuni de lucru cu registrele la nivel de octet, la nivel de bit și instrucțiuni de control și de încărcare imediată. Toate instrucțiunile durează un singur ciclu, ciclul de aducere fiind ascuns — adică suprapus peste execuția instrucțiunii precedente, cu excepția: testului condiționat dacă e adevărat sau cînd contorul program este modificat ca rezultat al unei operații. În aceste ultime cazuri durata vizibilă este de 2 cicluri mașină. Un ciclu mașină constă din 4 perioade ale oscilatorului pilot. Astfel la un oscilator pe 4 Mhz durata majorității instrucțiunilor este de 1 μs; la excepțiile evidențiate mai sus durata e de 2 μs.

În continuare, prin „f" am desemnat un registru din setul de registre și prin „d" destinația care dacă e 0 înseamnă că rezultatul e plasat în registrul temporar W și dacă e 1 rezultatul e plasat în registrul specificat „f".

Forma instrucțiunilor orientate pe octet este:

OPCODE	d	f
11 6	5	4 0

În instrucțiunile orientate pe bit „b" reprezintă bitul din registrul „f" afectat de operație. Forma acestor instrucțiuni este:

OPCODE	b	f
11 8	7 5	4 0

Pentru instrucțiunile de control sau de încărcare imediată k reprezintă o valoare de 8 (9) biți. Forma acestor instrucțiuni este:

OPCODE	k
11 8(9)	7(8) 0

Setul de instrucțiuni este prezentat în tabelul nr. 1.5.

Tabelul 1.5. SETUL DE INSTRUCȚIUNI AL PIC 16C5x

A) OPERAȚII CU REGISTRE ORIENTATE PE OCTET

Cod HEX	Nume	Mnemonica	Operanzi	Operația	Indic. afectați
000	No Operation	NOP		—	—
02f	Move W to f	MOVWF	f	W—>f	—
040	Clear W	CLRW		0—>W	Z
06f	Clear f	CLRF	f	0—>f	Z
08f	Substract W from f	SUBWF	f, d	f—W—>d [f+W++1—>d]	C, DC, Z
0Cf	Decrement f	DECF	f, d	f—1—>d	Z
10f	Inclusive OR W and f	IORWF	f, d	W v f—>d	Z
14f	AND W and f	ANDWF	f, d	W & f—>d	Z
18f	Exclusive OR W and f	XORWF	f, d	W ⊕ f—>d	Z
1Cf	Add W and f	ADDWF	f, d	W + F—>d	C,DC,Z
20f	Move f	MOVF	f, d	f—>d	Z
24f	Complement f	COMF	f, d	$\bar{f}$—>d	Z
28 f	Increment f	INCF	f, d	f+1—>d	Z
2Cf	Decrement f, Skip if zero	DECFSZ	f, d	f—1—>d skip if zero	—
30f	Rotate right f	RRF	f, d	f(n)—>d(n—1) f(0)—>C c—>d(7)	C
34f	Rotate left f	RLF	f, d	f(n)—>d(n+1) f(7)—>C C—>d(0)	C
38f	Swap halves f	SWAPF	f, d	f(0,—3)<—> f(4—7)—>d	—
3Cf	Increment f, Skip if zero	INCFSZ	f, d	f+1—> skip if zero	—

T(abelul 1.5. continuare)

B) OPERAȚII CU REGISTRE ORIENTATE PE BIT

Cod HEX	Nume	Mnemonica	Operanzi	Operația	Indic. afectați
4bf	Bit Clear f	BCF	f, d	0—>f(b)	—
5bf	Bit Set f	BSF	f, d	1—>f(b)	—
6bf	Bit Test f, Skip if Clear	BTFSC	f, d	Test bit (b) in file (f): Skip if Clear	—
7bf	Bit Test f, Skip if Set	BTFSS	f, d	Test bit (b) in file (f); Skip if set	—

C) OPERAȚII DE CONTROL ȘI IMEDIATE

Cod HEX	Nume	Mnemonica	Operanzi	Operația	Indic. afectați
002	Load OPTION register	OPTION	—	W—>OPTION register	—
003	Go into standby mode	SLEEP	—	0—>WDT, stop oscilator	TO, PD
004	Clear Watchdog timer	CLRWDT	—	0—>WDT (and presacaler, if assigned)	TO, PD
00f	Tristate port f	TRIS	f	W—>I/0 control register	—
8kk	Return place Literal in W	RETLW	k	k—>W Stack—>PC	—
9kk	Call subroutine	CALL	k	PC+1—>Stack k—>PC	—
Akk	Go To Address (k is 9 bit)	GOTO	k	k—>PC (9bits)	—
Ckk	Move Literal to W	MOVLW	k	k—>W	—
Dkk	Inclusive OR Literal and W	IORLW	k	k v W—>W	Z
Ekk	AND Literal and W	ANDLW	k	k & W—>W	Z
Fkk	Exclusive OR Literal and W	XORLW	k	k ⊕ W—>W	Z

INSTRUMENTELE DE DEZVOLTARE.

Firma producătoare, Microchip Technology Inc. oferă o serie de instrumente necesare proiectării aplicațiilor, cum ar fi: emulatoare, cross-asambloare, simulatoare și programatoare de EPROM. Instrumentele software sînt disponibile pe cele mai uzuale calculatoare între care și IBM PC, Apple Macintosh și DEC VAX. Emulatoarele, PICSIII, permit emularea în timp real a tuturor circuitelor din familie pînă la frecvența de 20 Mhz. Depanarea simbolică, oprirea pe instrucțiune, trasarea programului sînt cîteva din posibilitățile emulatorului dezvoltat pe IBM PC. Cross-asambloarele, PICAL-C, au printre facilități asamblarea condiționată și acceptarea macro-instrucțiunilor.

1.5. UTILITATEA VERSIUNILOR DE DEZVOLTARE

Versiunea de microcalculator integrat cu soclu pentru EPROM este comparabilă cu acele versiuni care au EPROM-ul integrat pe aceeași pastilă, cum este versiunea 8748 a microcalculatorului 8048 a firmei INTEL. Între aceste două versiuni de dezvoltare comparabile balanța pare să încline în favoarea variantei cu soclu. Avantajul constă în dezvoltarea mai multor programe în faza de prototip. Pe aceeași configurație hard se pot dezvolta simultan mai multe programe care se implantează în memorii EPROM separate. Numărul limitat de cicluri de scriere/ștergere a memoriilor EPROM nu afectează microcalculatorul integrat, cum e în cazul tipului 8748. Calitatea soclului este singurul lucru de natură să creeze rețineri față de versiunea de dezvoltare a firmei Zilog.

Datorită faptului că microcalculatorul Z8 nu dispune de un mod de lucru pas cu pas (single step) așa cum am văzut că are la INTEL 8048, sînt relativ greu de pus la punct programele. În această ipoteză chiar versiunea cu soclu de EPROM se poate dovedi ineficientă datorită modificărilor frecvente și numeroase ce trebuiesc operate în programe. Acum versiunea cu interfața de memorie se dovedește utilă întrucît memoria „adăugată“ poate fi la fel de bine o memorie EPROM dar și o memorie RAM cu dublu acces. Această memorie putînd face parte dintr-un microcalculator de dezvoltare (cu floppy

disc, sistem de operare adecvat, cross-asamblor de Z8 etc.) sub care se înscrie, se modifică şi se corectează programul Z8. După înscrierea programului dorit „felia“ de memorie în cauză se pune la dispoziţia exclusivă a microcalculatorului Z8 prin separarea de magistrala microcalculatorului de dezvoltare pe baza cererii de cedare a magistralei (BUSREQUEST) şi a confirmării acestei cedări (BUSACKNOWLDGE). Memoria în această accepţiune se numeşte simulator de EPROM.

2. STRUCTURA MICROCALCULATORULUI INTEGRAT Z8

2.1. CONEXIUNILE EXTERIOARE

Circuitele din familia Z8 au aceeaşi structură, diferenţele fiind minime. Diferenţele între diferitele circuite ale familiei constînd în faptul că memoria ROM este integrată sau nu, dacă capacitatea acestei memorii este de 2 Kocteţi respectiv de 4 Ko, dacă are sau nu opţiunea de consum redus. Din punct de vedere al setului de instrucţiuni nefiind nici o diferenţă. În figura 2.1 se prezintă schema bloc a circuitului Z8612 [UB 8810] care se regăseşte la toate celelalte circuite ale familiei.

Variantele de microcalculatoare Z8 destinate dezvoltării nu au integrată memoria ROM, avînd în schimb fie soclu pentru memorie fixă de tip EPROM fie magistrala de legătură a acesteia în exterior. Schema bloc a variantei de dezvoltare cu interfaţa pentru memoria program [UB 8820 sau Z8/64 — UB 8840] este redată în figura 2.2.

În figura 2.3 sînt prezentate conexiunile exterioare ale microcalculatoarelor integrate cu memoria ROM integrată. Aceste conexiuni în număr de 40, sînt următoarele: P00 — P07, P10 — P17, P20 — P27 şi P30 — P37 sînt liniile de intrare/ieşire compatibile TTL. Aceste 32 de linii de intrare/ieşire sînt grupate în 4 porturi de 8 biţi. Din punct de vedere funcţional aceste linii pot fi configurate într-o mulţime de moduri sub controlul programului. Liniile porturilor P1 şi P2 pot fi folosite pentru extinderea spaţiului de memorie (memoria externă). De asemenea pot fi puse în starea de impedanţă ridicată tot sub controlul programului. Numai liniile portului 2 pot fi configurate ca ieşiri open drain.

AS — este o linie de ieşire, activă pe zero o dată la fiecare ciclu de acces la memoria internă sau la memoria externă. Dacă nu sînt alte specificări, prin memoria externă a microcalculatorului integrat Z8 înţelegem, în continuare, atît memoria program externă cît şi

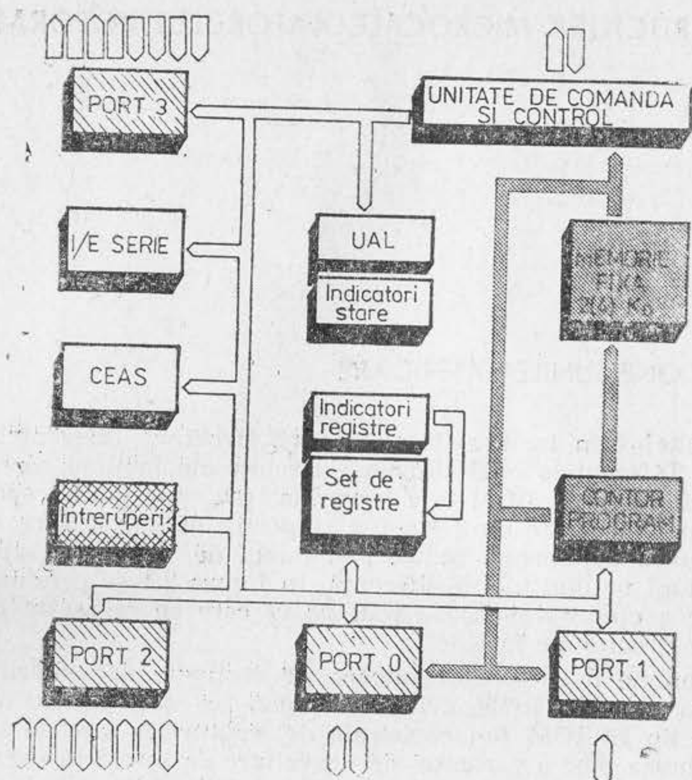


Fig. 2.1. Z8/40 [UB 8810]. Schema bloc

memoria externă de date. Adresa memoriei externe este validă pe frontul crescător al semnalului AS. Această adresă a memoriei externe apare pe liniile porturilor P0 și P1 după cum s-au configurat.

DS — este o linie de ieșire activă pe zero o dată pentru fiecare ciclu de transfer în/din memoria externă. În timpul unui ciclu de citire, data de la memoria externă este preluată prin portul P1 în timp ce linia DS este activă. În timpul unui ciclu de scriere în memoria externă, microcalculatorul Z8 plasează data (ce urmează să se înscrie în memoria externă) pe portul P1 în timp ce linia DS este activă. Cînd microcalculatorul integrat nu este configurat pentru memorie externă, linia DS devine activă pe o perioadă de tact ce precede începutul unui nou ciclu de aducere a codului operației din memoria program.

R/W — linie de ieșire activă pe zero cînd microcalculatorul Z8 scrie în memoria externă.

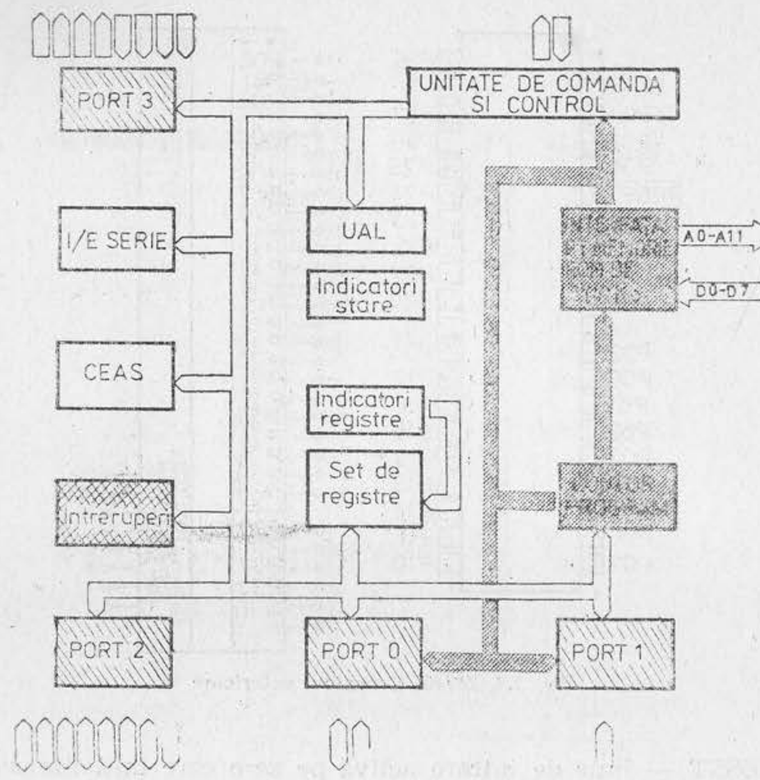


Fig. 2.2. Z8/64 [UB 8840]. Schema bloc

Liniile AS, DS și R/W împreună cu liniile porturilor P0 și P1 pot fi trecute în starea de impedanță ridicată. Liniile de control AS și DS sînt necesare datorită faptului că prin portul P1 sînt multiplexate atît adresele cît și datele. Linia R/W este necesară memoriei externe pentru a determina dacă ciclul de acces la ea este de citire sau scriere.

XTAL1, XTAL2 sînt linii de conexiune pentru un cristal de cuarț (maxim 8M Hz) sau o rețea LC respectiv RC. De asemenea se poate conecta direct tactul unui circuit de ceas exterior. În cazul variantelor de microcalculator integrat cu facilitate de consum redus tactul se generează extern și se conectează la microcalculatorul integrat pe linia XTAL1. La linia XTAL 2 este conectată sursa de alimentare pentru setul de registre interne și logica de inițializare în cazul căderii accidentale a tensiunii de alimentare Vcc.

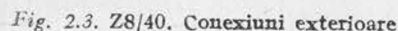


Fig. 2.3. **Z8/40. Conexiuni exterioare**

RESET — linie de intrare activă pe zero care inițializează microcalculatorul integrat Z8, care începe execuția programului de la adresa internă 0000CH. Linia de RESET trebuie ținută cel puțin 18 perioade de tact la zero pentru a inițializa microcalculatorul. După inițializare, o parte din registrele de control au o valoare determinată, iar liniile porturilor P0, P1 și P2 sînt trecute în modul de intrare.

Linia de RESET mai este folosită și în secvența de salvare a regiștrilor interni în cazul folosirii opțiunii de consum redus cît și în forțarea intrării în modul autotest. Acest din urmă mod este atins prin ridicarea tensiunii pe linia RESET la o valoare mai mare decît tensiunea de alimentare Vcc.

Versiunea de dezvoltare a microcalculatorului integrat Z8 are 64 de pini. Deosebirea față de versiunea standard constă în faptul că memoria internă ROM nu mai este integrată și liniile de adrese și date ale acestei memorii sînt amplificate și scoase la conexiunile exterioare. În figura 2.4 sînt prezentate conexiunile versiunii de dezvoltare Z8/64.

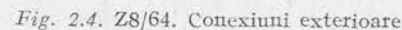


Fig. 2.4. Z8/64. Conexiuni exterioare

Funcțiile liniilor de control AS, DS, R/W, XTAL1, XTAL2, RESET și liniilor de I/E sînt identice cu cele din cazul versiunii standard. Funcțiile celorlalte linii vor fi descrise în continuare, astfel:

A0 — A11 — liniile de adresă a memoriei program (2 Kocteți respectiv 4 Kocteți.

D0 — D7 — liniile de date din primii 2 Kocteți (4 Kocteți) ai memoriei program.

MDS (Memory Data Strobe) — este o linie de ieșire activă pe zero în timpul unui acces la primii 2 Kocteți din memoria program.

SYNC — este o linie de ieșire activă pe zero pe durata unei perioade de clock ce precede ciclul de aducere a unei noi instrucțiuni.

SCLK (System Clock) — este tactul intern amplificat. Frecvența acestui tact este jumătate din frecvența cristalului de cuarț.

IACK (Interrupt Acknowledge) — linie de ieșire activă pe 1 pe perioada unui ciclu de întrerupere. Ciclul de întrerupere apare ca răspuns la o cerere de întrerupere.

2.2. SPAȚIUL DE MEMORIE

Microcalculatorul poate utiliza 3 tipuri de memorie:

— memoria program [2(4) Kocteți ROM intern respectiv 62(60) Kocteți ROM extern],

— memoria de date și

— regiștrii interni.

În figura 2.5 este prezentat spațiul de memorie al microcalculatorului integrat Z8 standard. În paranteză sînt trecute valorile de adrese ale microcalculatorului integrat cu 4 Kocteți memorie fixă integrată.

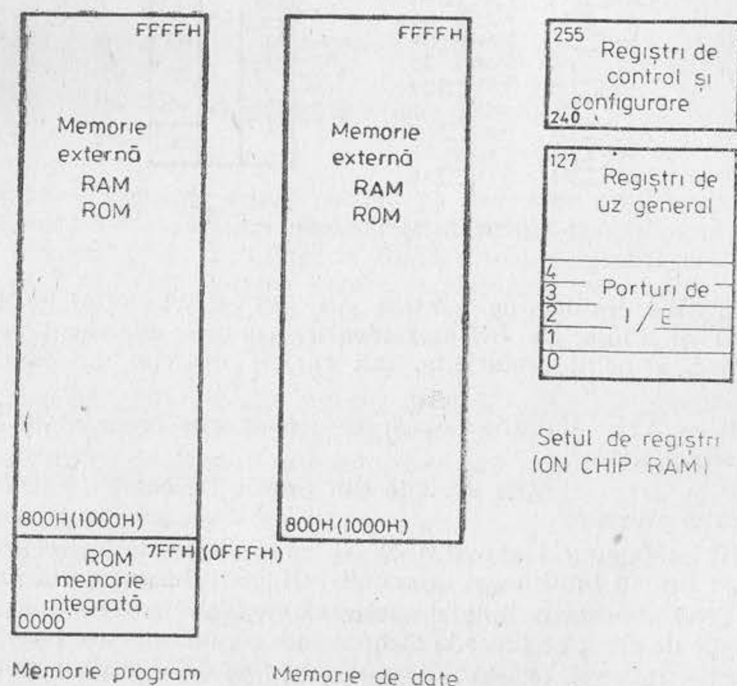


Fig. 2.5. Spațiu de memorie

2.2.1. MEMORIA PROGRAM

Memoria program este accesată printr-un contor program de 16 biți, deci capacitatea maximă a acesteia poate fi de 64 Ko. Memoria program este împărțită în 2 arii: memoria integrată de 2(4) Ko respectiv memoria program externă de 62(60) ko.

Pentru a accesa locațiile din memorie mai mari de 2047 (4095), microcalculatorul va efectua un ciclu de aducere a instrucțiunii următoare din memoria program externă. Pentru aceasta microcalculatorul trebuie apriori configurat corespunzător.

Primii 12 octeți din memoria program sînt rezervați vectorilor de întrerupere. Locațiile 0 — 0BH conțin 6 vectori de întrerupere pe 16 biți, vectori ce indică adresa subrutinelor de tratare a celor maxim 6 întreruperi mînuite odată de microcalculator. La inițializare în contorul program este forțată adresa 000CH, care este prima adresă disponibilă pentru programul utilizatorului. Mapa memoriei program este redată în figura 2.6. Este de remarcat faptul că la adresa n se află octetul superior al vectorului de întrerupere, iar la octetul $n+1$ se află octetul inferior al acestui vector, spre deosebire de procesoarele binecunoscute Intel 8080, Zilog Z80 etc.

Funcție de necesarul de memorie program, utilizatorul poate configura ușor calculatorul pentru a nu avea deloc memorie externă sau pentru a avea o memorie suplimentară de 256 octeți, de 2 Kocteți, sau de 62 Kocteți.

Pentru o memorie program suplimentară de 256 octeți [spațiul 2048 (800H) ÷ 2048 + 255 (8FFH)] se configurează portul P1 ca un port care multiplexează adrese și date (AD0 — AD7) care asigură adresele A0 — A7 pe durata semnalului, AS, respectiv preia datele D0 — D7 pe durata semnalului DS.

Pentru o memorie program suplimentară de 2 Kocteți (în cazul versiunii standard) pe lîngă P1 care se configurează așa cum s-a arătat mai sus se configurează semioctetul inferior al portului P0, P00 — P03, ca ieșire de adrese A8 — A11.

Pentru o memorie program suplimentară maximă se configurează întreg portul P0, P00 — P07, ca ieșire de adrese A8 — A15.

2.2.2. MEMORIA DE DATE

Microcalculatoarele integrate Z8 pot accesa o memorie de date externă de 62 (60) Kocteți. Modul de accesare a memoriei de date este identic cu cel pentru memoria program suplimentară (externă). Deosebirea între memoria de date și memoria program constă în aceea că memoria program se întinde pe întregul spațiu ocupat de

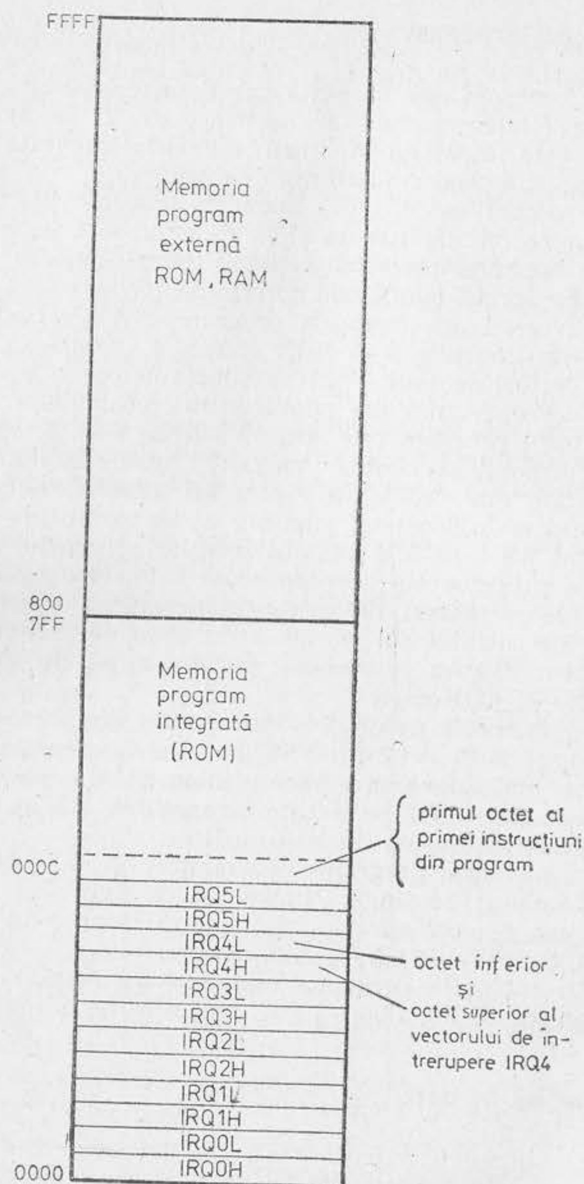
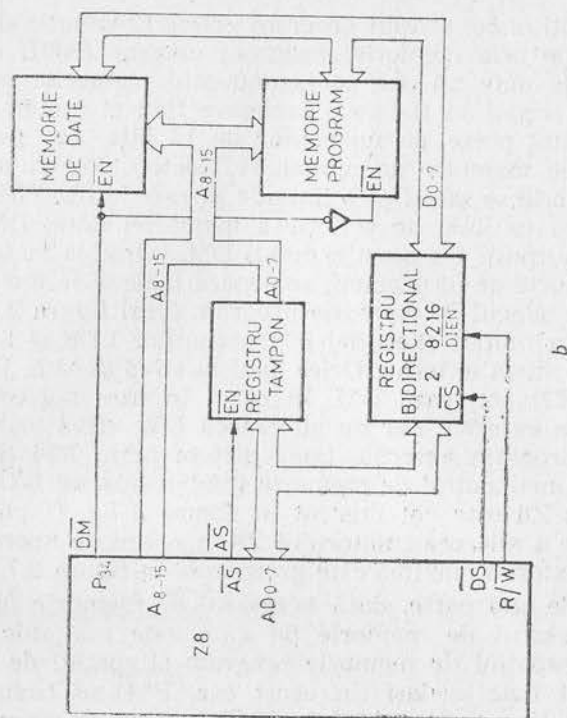


Fig. 2.6. Mapa memoriei program

cei 16 biţi ai contorului program pe cînd memoria de date se întinde numai pe aria memoriei program externe (800H — FFFFH). Memoria de date nu are corespondentul memoriei program integrate care ar trebui să fie nu o memorie fixă ci o memorie citeşte/scrie. Pe de altă parte, cu un contor de 16 biţi se poate adresa un spaţiu de memorie de numai 64 Kocteţi. Pentru mărirea spaţiului de memorie se sacrifică o linie de intrare/ieşire, P34. Aceasta se configurează ca linie de selecţie a memoriei date, DM (Data Memory Select output). Cu acest semnal, DM, introdus în selecţia unui banc de memorie de 62 Kocteţi, se separă bancul de memorie externă, de date, de bancul de memorie program (vezi figura 2.7). DM este activ numai în timpul execuţiei instrucţiunilor LDE şi LDEI sau în cazul folosirii stivei externe. Orice apel la stivă (CALL, PUSH, POP, RET sau IRET) activează DM. În cazul în care s-a configurat stiva în memoria externă, dar nu şi P34 ca DM, stiva va funcţiona în memoria program externă. Dacă privim pinul P34 (DM) ca o adresă, A16, atunci spaţiul de memorie maxim adresat de microcalculatorul integrat Z8 este cel ilustrat în figura 2.7.a. O posibilă schemă de utilizare a microcalculatorului Z8 în regim de microprocesor cu memorie externă maximă este prezentată în figura 2.7.b.

Pe de altă parte, dacă necesarul de memorie program împreună cu necesarul de memorie de date este mai mic sau egal cu 64 Kocteţi spaţiul de memorie program şi spaţiul de memorie de date poate fi fizic acelaşi. În acest caz, P34 nu trebuie configurat ca selecţie de memorie de date, DM. Înainte de prima referire la memoria externă, fie ea memoria de date fie ea memoria program, porturile P0, P1 şi P3 (P34) trebuiesc configurate corespunzător.

Din cauza suprapunerii execuţiei instrucţiunii precedente cu aducerea instrucţiunii curente (una din caracteristicile microcalculatorului integrat care îi asigură acestuia o viteză de execuţie mai mare) după ultima instrucţiune de configurare nu poate urma o instrucţiune care citeşte sau scrie în/din memoria externă. Evident, pentru că în timp ce se execută instrucţiunea de configurare s-ar aduce din memoria program următoarea instrucţiune. Ori în timpul execuţiei instrucţiunii de configurare a unuia din porturile P0 sau P1 acestea nu pot fi folosite pentru adresarea, în acelaşi timp, a unei instrucţiuni sau a unei date din memoria externă. Deşi s-a menţionat această restricţie din punct de vedere practic se poate evita uşor introducînd două instrucţiuni de un octet sau una de doi octeţi, instrucţiuni utile programului sau instrucţiuni care nu deranjează programul (ca de exemplu două instrucţiuni NOP). Programul utilizator înscris în microcalculatorul integrat poate începe cu configurarea porturilor P0, P1 şi P3 după care se fixează stiva şi se pro-



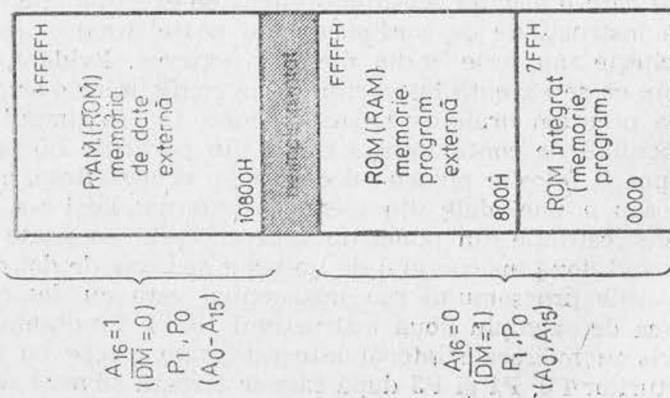


Fig. 2.7. Spaţiu de memorie maxim adresabil

gramează portul P2. După această secvenţă poate urma orice instrucţiune, inclusiv referitoare la memoria externă, evitîndu-se astfel orice consecinţe negative ce ar decurge din restricţia menţionată.

2.2.3. SETUL DE REGISTRE INTERNE

Memoria citeşte/scrie de 144 octeţi este de fapt un set de registre interne. Cele 144 de registre cuprind:

— 4 registre corespunzătoare porturilor P0 — P3 (R0—R3),
— 124 registre de uz general R4 — R128 şi
— 16 registre de control şi stare (R127 — R255)

Se observă că între registrul R127 şi R240 nu este implementată memorie, fapt reflectat şi în setul de instrucţiuni. În tabelul 2.1 sînt prezentate toate registrele interne cu numele, identificatorii şi adresele lor.

Aceste registre sînt citite cînd sînt adresate ca sursă în cadrul unei instrucţiuni şi sînt înscrise cînd sînt adresate ca destinaţie.

Registrele de control şi configurare şi registrele porturilor de I/E au fost incluse în setul de registre al microcalculatorului integrat cu scopul de a permite instrucţiunilor să proceseze intrările şi ieşirile fără să fie nevoie de instrucţiuni speciale de intrare/ieşire. Pentru a ilustra această afirmaţie considerăm un exemplu simplu: pe un port de ieşire paralelă la un moment dat vrem să punem la zero biţi 0, 2, 4 şi 7 (01101010 binar — 6A hexa). În cazul lui Z8 aceasta se realizează executînd o singură instrucţiune:

AND P0, 6A H,

în timp ce în cazul procesoarelor 8080 sau Z80 acelaşi lucru se realizează cu următoarea secvenţă:

```
INP A, P0
AND 6AH
OUT P0, A.
```

În general toate registrele pot funcţiona ca acumulator, indicator de adresă sau registru de index. Accesul la registre se poate face direct sau indirect printr-un cîmp de adresă de 8 biţi. Pentru reducerea spaţiului de memorie şi a timpului de execuţie, microcalculatorul integrat, Z8, dispune de un mecanism de adresare eficient bazat pe pointerul de registre RP (R253 — FDH). Acest mecanism de adresare îl vom numi în continuare şi mod de adresare pe 4 biţi. Deoarece pentru adresarea unui registru este nevoie de 8 biţi de

Tabelul 2.1.

Locaţia		Reg. de control şi configurare	Identificatori
FF	255	indicator de stivă (0 — 7)	SPL
FE	254	indicator de stivă (8 — 15)	SPH
FD	253	pointer de registre	RP
FC	252	indicatori de stare	FLAGS
FB	251	reg. de mascare a întreruperilor	IMR
FA	250	reg. cererilor de întrerupere	IRQ
F9	249	reg. de priorităţi a întreruperilor	IPR
F8	248	reg. de configurare a port. P0, P1	P01M
F7	247	reg. de configurare a port. P3	P3M
F6	246	reg. de configurare a port. P2	P2M
F5	245	predivizor numărător T0	PRE0
F4	244	numărător T0	T0
F3	243	predivizor numărător T1	PRE1
F2	242	numărător T1	T1
F1	241	reg. de configurare numărătoare	TMR
F0	240	reg. de transmisie/recepţie serie	SI0
ZONA NEIMPLEMENTATĂ			
7F	127	registre de uz general	R127
04	004		R4
03	003	portul P3	P3
02	002	portul P2	P2
01	001	portul P1	P1
00	000	portul P0	P0

adresă, adresarea pe 4 biţi constă în utilizarea pointerului de registre RP care conţine semioctetul superior al adresei; semioctetul inferior fiind furnizat de instrucţiunea care apelează registrul în cauză. Pentru aceasta setul de registre interne s-a împărţit în 9 grupe, fiecare grup fiind compus din 16 registre contigue. Registrele dintr-o grupă se mai numesc registre de lucru. Acest mecanism este util, scurtînd codul şi reducînd timpul de execuţie, în cazul în care se lucrează în mod repetat într-o zonă (grupă) a setului de registre. De exemplu la iniţializare cînd se configurează microcalculatorul integrat şi trebuiesc înscrise registrele de control şi configurare, R240 — R255, se înscrie în RP (R253) adresa ultimului grup de registre F0 hexa ca apoi adresarea celorlalte registre de control şi configurare din grup să se facă numai specificînd semioctetul inferior. Acest mecanism de adresare este prezentat şi grafic în figura 2.8.

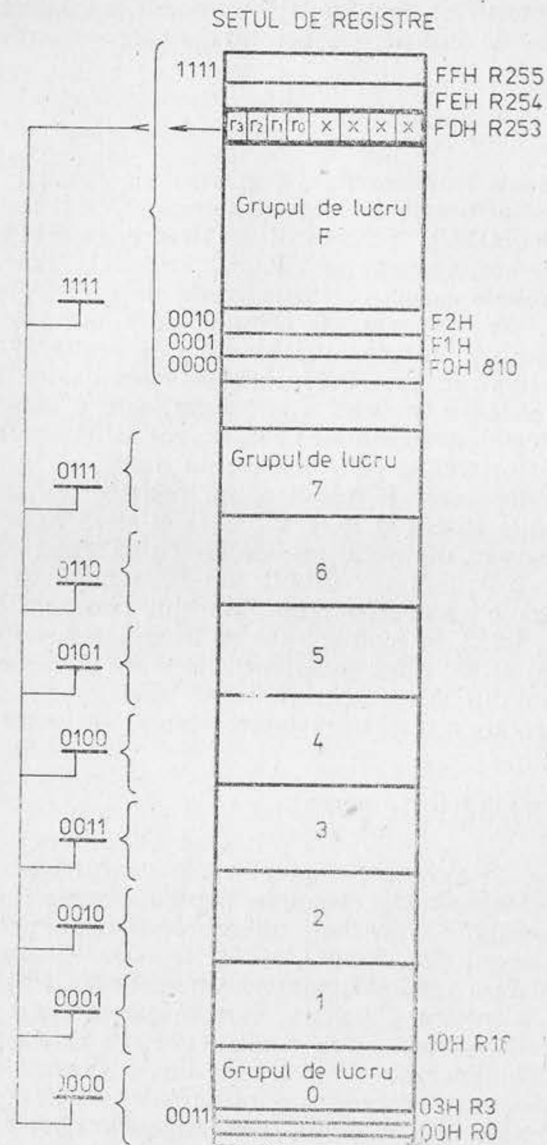


Fig. 2.8. Adresarea regiştrilor prin registru indicator RP

Orice instrucţiune care poate înscrie registrele generale poate înscrie şi pointerul de registre, RP, existînd în plus şi o instrucţiune specializată de înscriere a acestui pointer — anume SRP.

2.2.4. STIVA

Stiva se poate implementa fie în setul de registre interne fie în memoria externă funcţie de bitul de selecţie D2 din registrul de configurare R248 (PO1M). Indicatorul de stivă are 16 biţi pentru cazul utilizării memoriei externe sau 8 biţi în cazul folosirii registrelor interne. În ambele cazuri există restricţii de care trebuie să se ţină seama. Stiva externă poate fi oriunde între adresele 800 hexa şi FFFF hexa iar cea internă începînd de la registrul R127 pînă la R4. Adresa stivei se ţine în registrele specializate R255 (SPL) şi R254 (SPH). Evident în cazul stivei interne se utilizează numai registrul R255, celălalt nu mai are nici un rol în determinarea punctului de intrare în stivă şi prin urmare nu numai că nu trebuie să fie înscris cu 00 dar poate fi folosit ca un registru de uz general.

Instrucţiunile PUSH şi POP salvează în stivă respectiv refac din stivă orice registru din setul registrelor cu excepţia registrelor care nu se citesc. Instrucţiunea CALL sau recunoaşterea unei întreruperi debutează cu salvarea numărătorului program şi a indicatorilor de stare, R252, în mod similar cu instrucţiunea PUSH. Instrucţiunile RET şi IRET refac indicatorii de stare şi valorile numărătorului program din stivă la terminarea unei subrutine sau a unei secvenţe de tratare a unei întreruperi (similar cu instrucţiunea POP).

2.3. DIAGRAMELE DE TIMP

Diagramele de timp a semnalelor din cadrul logicii tradiţionale (cablate) secvenţiale sînt esenţiale pentru înţelegerea acestor circuite. Comparativ cu acestea, microprocesoarele prin cele cîteva semnale de control (MI, I/ORQ, MEMRQ, RD, WR sau I/OW, I/OR, MEMR, MEMW) a resurselor externe (memorii, CI/E) sînt mai uşor de controlat şi înţeles. Ca atare, complexitatea fenomenelor de interfaţare e mult redusă, lucru confirmat şi de cele cîteva diagrame de timp a semnalelor de control care sînt suficiente pentru înţelegerea unui sistem microprocesor indiferent de complexitate. În cazul microcalculatorului integrat, care înglobează tot ce este necesar unui sistem de calcul de sine stătător, nu mai apare nevoia semnalelor de control şi prin urmare nici studierea desfăşurării acestora în timp. Toate acestea fiind spuse, evident din punctul de vedere al

utilizatorului logicii secvenţiale, microprocesorului, respectiv al microcalculatorului integrat.

Datorită faptului, particular, că microcalculatorul integrat la care ne referim, Z8, a fost proiectat astfel încît să poată fi configurat pe lîngă microcalculatorul integrat propriu-zis (sau de sine stătător) şi ca microprocesor de uz general, este necesar să cunoaştem diagramele de timp ale semnalelor de control. Un alt motiv ar fi existenţa versiunii de dezvoltare, cu magistrala internă de conectare a memoriei program scoasă în afară pentru substituirea acesteia cu o memorie EPROM.

Formele de undă care urmează se dau în raport cu tactul extern (8MHz), sau cu tactul intern (4MHz) şi cu ciclurile maşină (0,75 µs).

2.3.1. SUPRAPUNEREA INSTRUCŢIUNILOR

Viteza de execuţie mare se datorează atît frecvenţei interne mari (4MHz) cît şi suprapunerii perioadei de execuţie a unei instrucţiuni cu perioada de aducere din memoria program a următoarei instrucţiuni.

În literatura de specialitate [1], [2] credem că se greşeşte puţin vorbindu-se de „instructions pipelining". Cu toate acestea trebuie recunoscut că deşi nu există o coadă de aşteptare la execuţia instrucţiunilor se poate considera Z8 avînd o arhitectură secvenţial-paralelă. Funcţionarea în paralel a secţiunii care se ocupă de aducerea unei noi instrucţiuni cu execuţia propriu-zisă a instrucţiunii curente se datorează existenţei a două magistrale interne (v. fig. 2.9.).

Această funcţionare este transparentă utilizatorului cu excepţia cazurilor cînd apar ramificaţii în program sau cînd se fac referiri la memoria externă. Instrucţiunile în care nu are loc suprapunerea pe execuţia lor a aducerii instrucţiunii următoare sînt următoarele: CALL, JP, LDC1, LDC, LDE, LDEI, RET, IRET şi NOP. În aceste cazuri (cu excepţia instrucţiunii NOP) numărătorul program primeşte o nouă adresă de la secţiunea de execuţie prin magistrala proprie. Pentru moment secţiunea de execuţie aşteaptă extragerea noii instrucţiuni. După ce a fost extrasă prima instrucţiune, contorul program împreună cu magistrala proprie intră în regim obişnuit în care după extragerea unei instrucţiuni începe extragerea următoarei. În acest fel execuţia propriu-zisă devine transparentă. În figura nr. 2.10 este ilustrat modul de suprapunere a instrucţiunilor.

În continuare vom numi „ciclu efectiv" partea din timpul total de execuţie a instrucţiunii în care are loc aducerea instrucţiunii for-

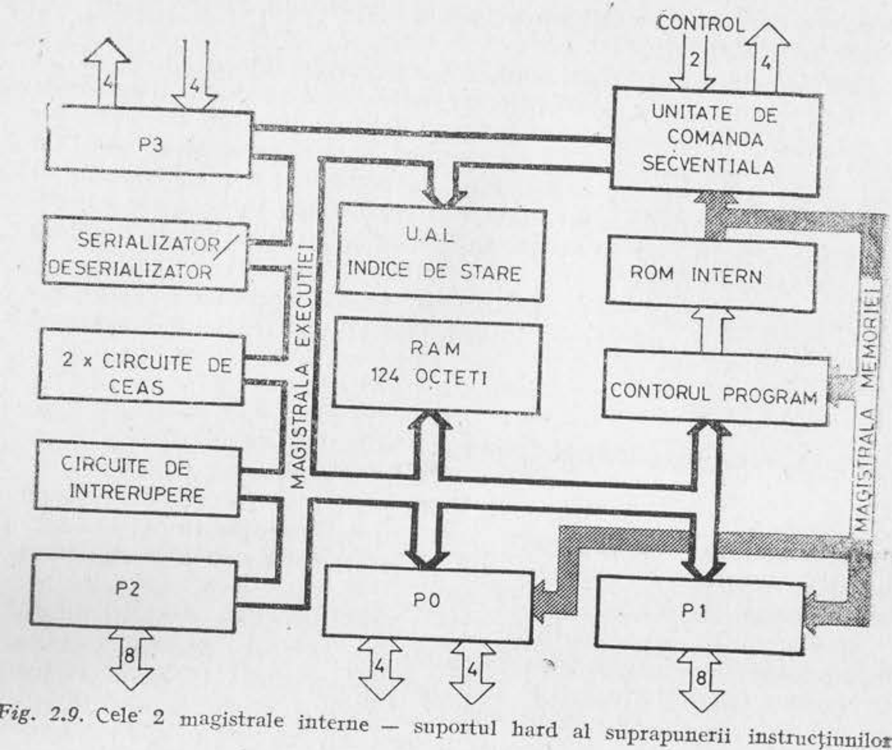


Fig. 2.9. Cele 2 magistrale interne — suportul hard al suprapunerii instrucţiunilor

mată din 1, 2 sau 3 octeţi fiecare. Ciclul ascuns este partea de timp din instrucţiune în care are loc procesul extragerii următoarei instrucţiuni. Cînd calculăm timpul de execuţie a unei instrucţiuni dintr-un program trebuie să luăm în calcul numai ciclul efectiv aşa cum a fost definit mai sus şi cum e dat la fiecare instrucţiune în parte. Deasemenea, cînd se calculează durata unor programe de test (benchmark) se ia în calcul numai acest timp efectiv.

2.3.2. ADUCEREA INSTRUCŢIUNILOR DIN MEMORIA PROGRAM

Cînd este adresată memoria externă, porturile P0 şi P1 trebuiesc configurate adecvat. Portul P1 este utilizat pentru vehicularea multiplexată a octetului inferior de adresa (A0—A7) şi a octetului de date (D0—D7). Liniile portului P1 se vor nota în acest caz şi AD0—AD7. Portul P0 este folosit pentru octetul superior de adresa, A8—A15 sau numai A8—A11).

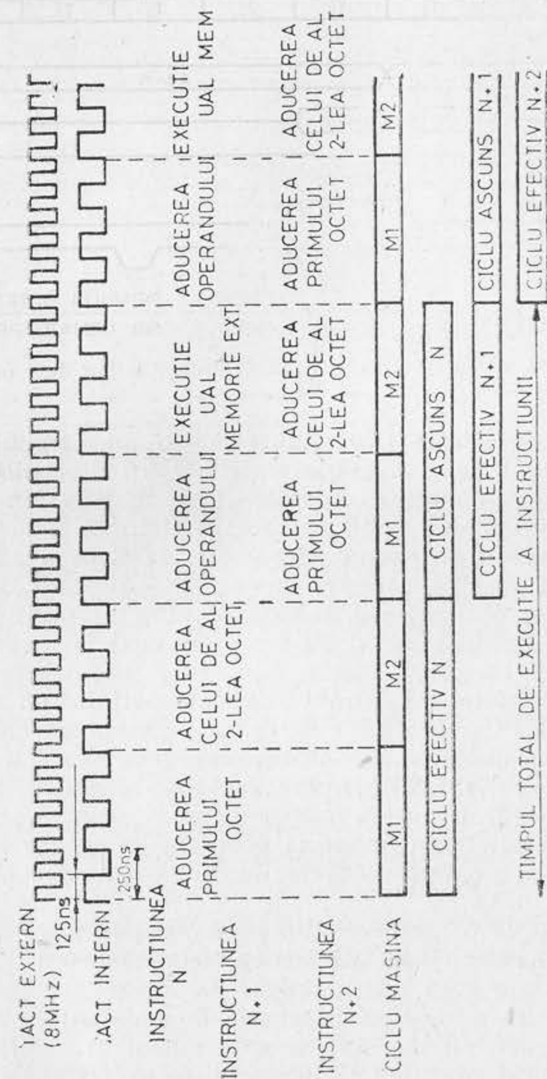


Fig. 2.10. Suprapunerea instrucţiunilor

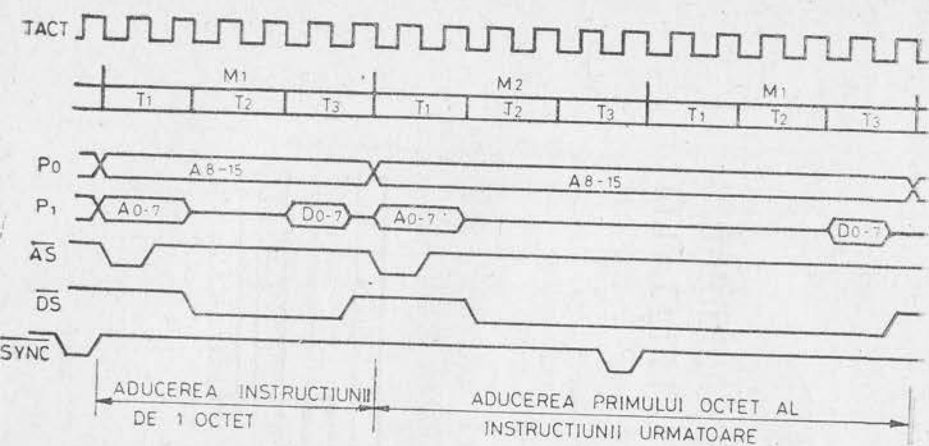


Fig. 2.11. Ciclul de aducere în cazul instrucțiunilor de 1 octet

Magistrala de adrese (A0—A7 și A8—A15), strobul de adrese (AS), și semnalul de citire/scriere (R/W) sînt disponibile la începutul fiecărui ciclu mașină. Adresele puse în exterior prin portul P0 rămîn stabile pe tot ciclul mașină, în timp ce semioctetul inferior, A0—A7, scos prin portul P1 rămîne stabil pe durata primului tact al ciclului mașină. Aceste adrese, A0—A7, sînt garantate pe frontul crescător al strobului, AS. După sfîrșitul primului tact din ciclul mașină (Mm T1) portul P1 trece automat în modul de intrare. Începînd cu tactul al doilea al ciclului mașină (Mm T2) este activ strobul de date, DS. Strobul de date permite plasarea pe intrările portului P1 a octetului de date. Unitatea de control și comandă a microcalculatorului Z8 acceptă data, via P1, pe frontul crescător al strobului, DS, în timpul celui de-al treilea tact al ciclului mașină (Mn T3). Cu aceasta se încheie un ciclu mașină, ciclu de aducere a unui octet din memoria program [fetch cycle]. În figura nr. 2.11 sînt prezentate formele de undă corespunzătoare unei instrucțiuni de un octet.

Aici se observă o „imperfecțiune" a mecanismului de suprapunere a instrucțiunilor. Dacă o instrucțiune de 2 octeți durează două cicluri, atunci faptul că instrucțiunea de 1 octet durează tot două cicluri naște niște semne de întrebare. Considerentele care au dus la această „imperfecțiune" sînt desigur numai la îndemîna proiectantului microcalculatorului Z8. Un puls de sincronizare, pe fiecare instrucțiune, este furnizat pe durata unui tact intern, înainte de începerea primului ciclu de aducere a noii instrucțiuni (M1). Acest puls este denumit SYNC, este activ pe zero și este scos în afară la

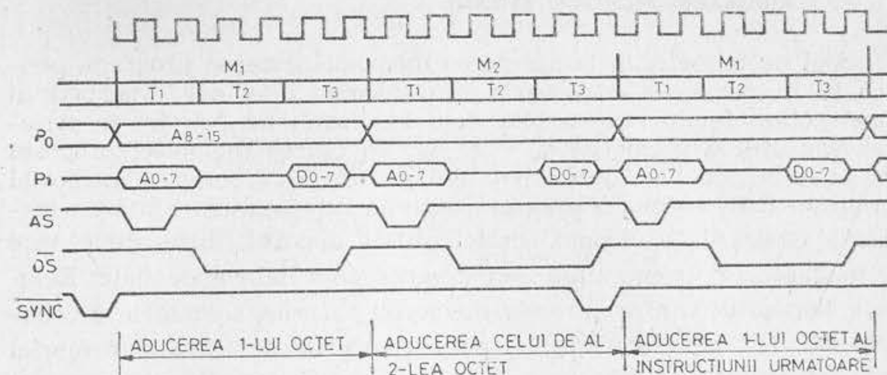


Fig. 2.12. Ciclul de aducere în cazul instrucțiunilor de 2 octați

versiunile Z8/64 pe un pin omonim iar în cazul versiunii standard acest puls este scos în afară pe pinul DS numai dacă nu se face acces la memoria externă.

În figura nr. 2.11 se prezintă succesiunea în timp a adreselor, semnalelor de strob AS și DS etc. pentru cazul unei instrucțiuni de 2 octeți. Cazul instrucțiunilor de 3 octeți nu a mai fost prezentat grafic întrucît este similar cu cel prezentat pentru instrucțiunile de 2 octeți; ciclul de aducere a instrucțiunii următoare are loc pe ciclul mașină M4 a instrucțiunii de 3 octeți, adică pe primul ciclu de execuție propriu-zisă a instrucțiunii de 3 octeți.

Trebuie spus că ciclurile de aducere a instrucțiunilor din memoria program sînt identice indiferent dacă adresele de la care se aduc instrucțiunile sînt mai mici de 0800H(2048) sau mai mari de 07FFH(2047); adică indiferent dacă memoria program adresată este internă sau externă. În plus, dacă microcalculatorul integrat este configurat pentru memoria externă dar se adresează pe moment memoria program internă, adresele sînt totuși scoase afară via P0, respectiv P1; în schimb, semnalele DS și R/W rămîn inactive.

Dacă microcalculatorul integrat este configurat pentru regimul de sine stătător atunci linia R/W rămîne inactivă iar pe ieșirea DS este scos semnalul de sincronizare SYNC. Versiunea de dezvoltare a microcalculatorului integrat, Z8/64, avînd un pin destinat acestui semnal, pentru situația amintită atît ieșirea de R/W cît și DS rămîn inactive.

2.3.3. ADRESAREA MEMORIEI EXTERNE

Aici nu ne referim la adresarea memoriei externe program, pentru că în acest caz ar fi vorba de aducerea unei noi instrucțiuni (instruction fetch) care a fost deja analizată; ne referim la scrierea sau citirea memoriei de date sau la citirea memoriei program ca un tabel de date (look up table) sau chiar la scrierea memoriei program dacă memoria program externă este implementată cu memoria citește/scrie. Scopul acestei ultime operații, între altele, este autoadaptarea programului sau crearea unor tabele de date. Relațiile temporale dintre adresele memoriei externe, semnalele de control AS, DS, R/W și DM(P34) pentru cazurile de citire a memoriei sînt prezentate în figura nr. 2.13 iar pentru cazurile de scriere a memoriei în figura nr. 2.14.

Bineînțeles că și în cazurile acestea, specificate încă de la început, accesarea memoriei externe se face în același mod, descris anterior, folosind porturile P0 pentru semioctetul superior de adrese, A8—A15, iar portul P1 multiplexat pentru adrese și date A/D0—A/D7. Adresele, A0—A15, sînt valide pe frontul crescător al strobului de date AS pentru ambele situații de scriere, respectiv citire. Deoarece portul P1 este multiplexat, adresele A0—A7 trebuiesc strobate (zăvorite) într-un registru.

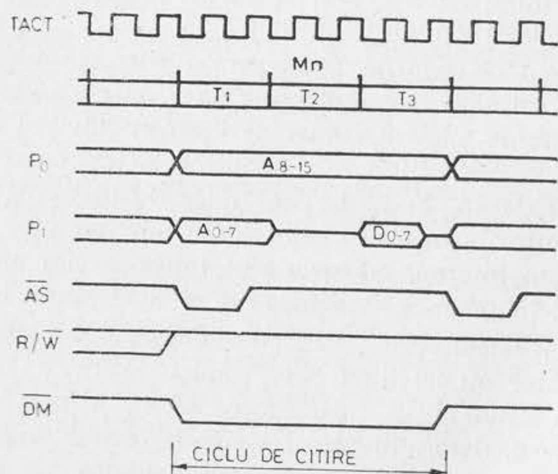


Fig. 2.13. Citirea memoriei externe

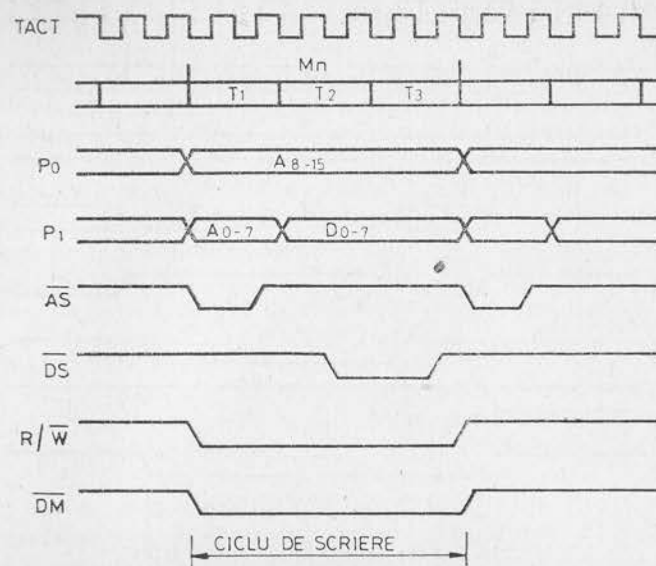


Fig. 2.14. Scrierea în memoria externă

În timpul ciclului de citire, datele de intrare trebuie să fie valide la intrarea portului P1 pe frontul pozitiv al strobului DS.

În timpul ciclului de scriere, adresele se încadrează în aceleași reguli amintite la ciclurile de aducere a instrucțiunilor sau de citire. Datele ce urmează a fi scrise în exterior sînt valide pe întreaga perioadă a strobului DS iar semnalul R/W este activ (zero) pe toată durata ciclului de scriere.

Semnalul, DM, de selecție a bancului memoriei de date furnizat pe portul P3 bitul 4 (P34) este activ pe toată durata ciclului, dacă a fost selectată memoria de date. În figurile nr. 2.15 și 2.16 se prezintă citirea respectiv scrierea în, sau dintr-o memorie externă cu deosebirea, față de cazurile prezentate în figurile 2.13 și 2.14, că s-a introdus prin programare o perioadă de tact suplimentară, notată TW. Aceasta este o facilitate prin care se poate face extinderea de memorie și pe seama unor componente mai lente. Programarea acestei stări de așteptare suplimentare se face prin registrul de configurare și control, POIM, (R248) bitul D5.

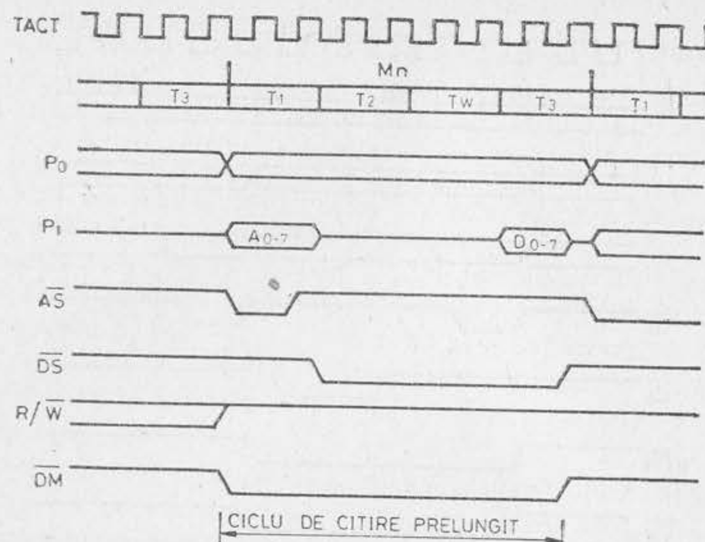


Fig. 2.15. Citirea memoriilor externe lente

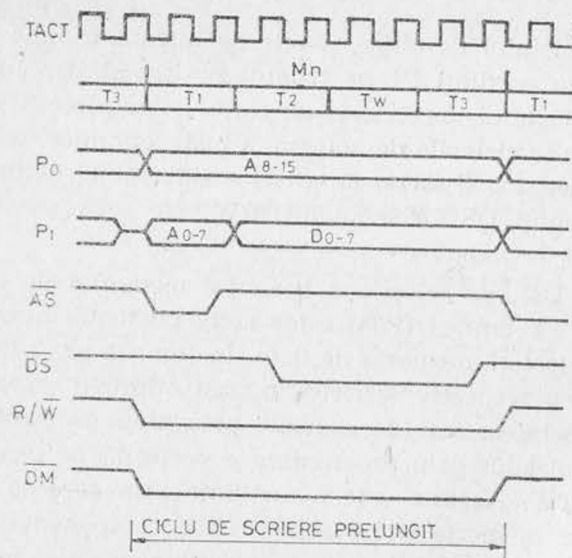


Fig. 2.16. Scrierea în memoriile externe lente

2.4. PORTURILE DE INTRARE/IEȘIRE

Cele 4 porturi de intrare/ieșire au fiecare cîte 8 linii. Aceste linii pot fi configurate funcție de aplicație ca:

— linii de intrare
— linii de ieșire
— linii de adrese
— linii de adrese și date
— linii de stare și control
— linii de interfață serie
— linii de intrare/ieșire pentru ceas
— linii de protocol de interfață paralelă.

Toate cele 32 de linii sînt compatibile TTL.

Prin configurare se înțelege alegerea din structura multifuncțională implementată hardware a acelei părți necesare unei funcții dorite. Alegerea se face prin program și constă din încărcarea corespunzătoare a regiștrilor de configurare și control. Toți regiștrii de configurare și control sînt dedicați unor anumite funcții legate de circuitele de intrare/ieșire ale microcalculatorului.

Scrierea respectiv citirea porturilor de intrare/ieșire se face folosind alți regiștri dedicați numiți regiștri de intrare/ieșire ai porturilor P0, P1, P2 și P3. Scrierea/citirea circuitelor de ceas, de interfața serie se face prin regiștrii de control și configurare dedicați, alții decît regiștrii de configurare efectivi. De exemplu, prin R247 [P3M] se dedică liniile P30 și P37 comunicației serie, se activează circuitul de interfață serială. Transmiterea respectiv recepționarea datelor serie se face tot printr-un alt registru de configurație și control, și anume R240[SIO].

2.4.1. STRUCTURA PORTURILOR DE INTRARE/IEȘIRE PORTURILE P0, P1, P2 ȘI P3

Porturile P0, P1 și P2 avînd aceeași structură vor fi tratate împreună. Fiecare din aceste porturi are o secțiune de intrare, o secțiune de ieșire și o logică de control comună. Pe partea de intrare aceste porturi au un registru tampon (input buffer) și un registru (input register). Similar, pe cealaltă secțiune există registru tampon de ieșire (output buffer) și registru de ieșire (output register).

Toate aceste patru registre enumerate sînt de cîte 8 biți corespunzător celor 8 linii ale portului. Diagrama bloc generală a acestor porturi este ilustrată în figura nr. 2.17.

Cînd un bit din aceste porturi este configurat ca ieșire, dacă scriem o valoare în acest bit, respectiva valoare 0 sau 1 se regăsește în registrul de ieșire dar și în registrul tampon de ieșire

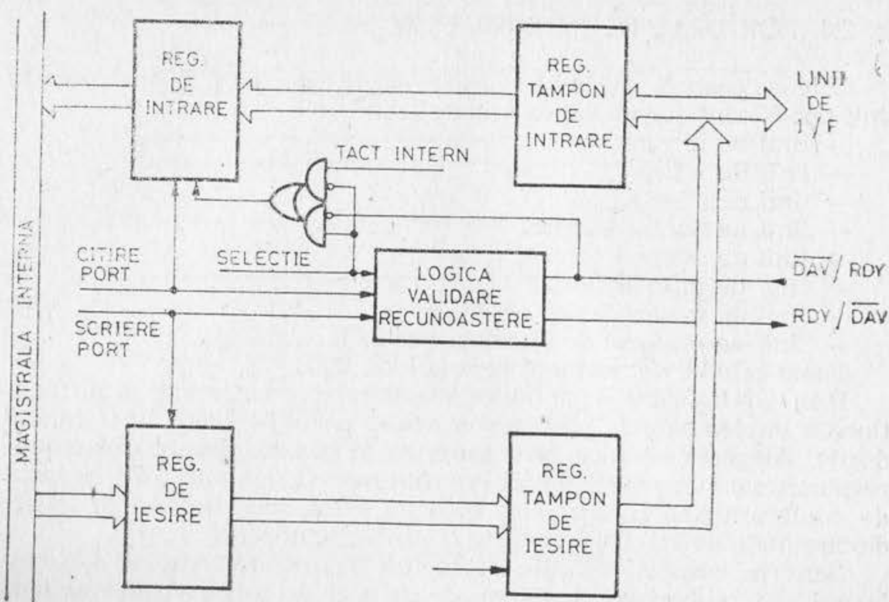


Fig. 2.17. Porturile P0, P1 și P2. Diagrama bloc.

și prin urmare și pe linia de ieșire corespunzătoare. Aceeași valoare se regăsește și pe poziția corespunzătoare a registrului tampon de intrare. Prin urmare, dacă citim un port de ieșire regăsim în valoarea citită, valoarea scrisă în port. Acest lucru este valabil cu o excepție care se referă numai la portul P2 în varianta de etaj final de ieșire cu drena în gol (open drain).

Cînd unul din porturile analizate este configurat ca intrare, citirea respectivului port înseamnă depunerea la „destinație" a valorii prezente pe liniile de intrare. Dacă portul de intrare este însoțit de semnale de control (validare și recunoaștere — handshake) la citirea respectivului port, valoarea depusă la destinație este valoarea zăvorîtă în registrul de intrare de strobul de intrare (DAV — Data available) și nu valoarea de la intrarea portului în momentul citirii. În cazul acesta, al portului configurat ca intrare cu semnale de control, portul poate fi înscris, iar valoarea va fi înmagazinată în registrul de ieșire. Evident această valoare nu mai poate fi citită înapoi. Acum, dacă ulterior înscrierii unui port configurat ca intrare, se reconfigurează ca ieșire, valoarea din registrul de ieșire se va regăsi pe liniile externe. Acest fapt permite o inițializare elegantă în sensul că la activarea (configurarea) portului de ieșire pe sarcinile externe se va găsi valoarea dorită fără să treacă prin alte

valori. În acest context, trebuie spus că aceasta este posibilă nu numai datorită structurii portului de intrare/ieșire ci și datorită faptului că la punerea sub tensiune sau după aplicarea unui semnal de „RESET" porturile de I/E se vor găsi în starea de intrare.

Din punct de vedere al structurii, portul P3 se deosebește de celelalte prin faptul că este împărțit în două, fixe din punct de vedere al sensului de vehiculare a datelor: 4 linii de intrare P30—P33 și 4 linii de ieșire P34—P37. Diagrama bloc a acestui port este redată în fig. 2.18. Liniile de intrare P30—P33, trecute prin registrul de intrare (de 4 biți) trec spre magistrala internă dar și spre circuitul de ceas, circuitele de tratare a întreruperilor și spre logica de control a celorlalte porturi de intrare/ieșire.

Liniile de ieșire, P34—P37, sînt servite de un registru de ieșire, de un registru tampon și de un registru de citire a datelor din registrul de ieșire, toate avînd o lungime de 4 biți.

Registrul de ieșire poate fi scris direct printr-o instrucțiune, sau indirect prin programarea corespunzătoare a circuitului de ceas, circuitului de serializare/deserializare sau a semnalelor de control a interfețelor.

La citirea portului P3, valoarea preluată de destinație se compune din valoarea momentană de pe liniile de intrare (P30—P33)

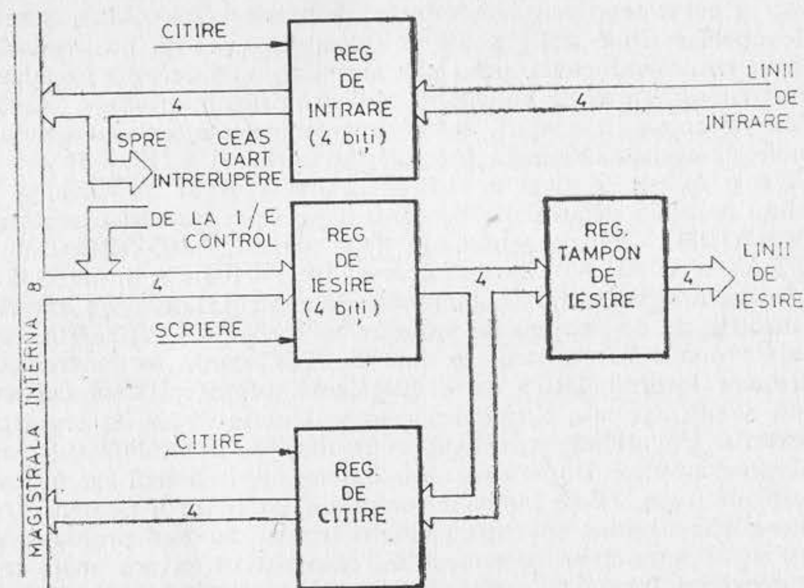


Fig. 2.18. Portul P3. Diagrama bloc

şi din data stocată în registrul de ieşire şi nu din valorile prezente pe linii de ieşire (P34—P37).

Registrul de ieşire al portului P3 nu poate fi scris dacă este folosit în conjuncţie cu circuitul de intrare/ieşire serie, cu circuitul de ceas sau cu semnale de control de interfaţă.

2.4.2. PORTUL P0

Portul P0 poate fi configurat fie ca port de intrare sau ieşire sau intrare şi ieşire, fie ca port de adresă utilizat în adresarea memoriei externe sau a unor circuite de intrare ieşire suplimentare. O altă particularitate a acestui port este posibilitatea de a-l programa diferit pe fiecare semioctet. Configurarea se face programînd registrul R248(P01M). Dacă portul P0 (sau numai semioctetul superior) este folosit ca port de I/E accesul se face prin registrul dedicat R0. În cazul portului de ieşire, acesta se scrie specificînd registrul R0 ca registru de destinaţie al unei instrucţiuni.

În cazul portului ca intrare, data din exterior este citită specificînd registrul R0 ca registru sursă a unei instrucţiuni. Semioctetul sau întregul port dacă e definit ca adresă nu poate fi citit sau scris ca registru. Funcţie de necesarul de memorie externă se poate utiliza în acest scop numai semioctetul inferior (A8—A11), semioctetul superior (P04—P07) poate fi folosit ca port de intrare/ieşire. În cazul cînd semiocteţii, unul sau amîndoi, sînt definiţi ca adrese, se pot trece în starea de impedanţă ridicată prin programare (R248—P01M). Trecerea în această stare se face concomitent cu celelalte semnale de control necesare memoriei externe AS, DS şi R/W.

Cînd e folosit ca port de I/E se poate folosi şi în varianta de interfaţă paralelă cu semnale de control prin programarea registrului R247(P3M). Pentru semnalele de control, RDY(READY) şi DAV(DATA AVAILABLE) se folosesc din portul P3 liniile P32 şi P35. În regimul de intrare cu semnale de control datele din exterior sînt însoţite de un semnal de validare ce intră pe P32(DAV0), care zăvorăşte datele în registrul de intrare. Tot acuma se generează o întrerupere internă, IRQ0 care, autorizată, oferă mijlocul cel mai elegant şi eficace de a sincroniza microcalculatorul Z8 cu dispozitivul extern. Un alt mijloc constă în explorarea liniei P32 sub controlul programului. După zăvorîrea datelor în registrul de intrare, amintită mai sus, Z8-ul pune la zero linia P35(RDY0) pentru a comunica dispozitivului extern că datele trimise au fost preluate. Pe baza acestui semnal de recunoaştere, dispozitivul extern poate termina secvenţa începută, ridicînd semnalul de validare DAV la 1 logic pentru a putea începe o nouă secvenţă de transfer. O nouă

secvenţă de transfer de date de la dispozitivul extern la Z8 trebuie să se desfăşoare numai după ce Z8-ul confirmă prin semnalul RDY că e gata să primească o nouă dată. Z8-ul e gata de un nou transfer numai după citirea datei tocmai zăvorîte în registrul de intrare. Citirea datei constă în specificarea registrului P0 ca sursă într-o instrucţiune. După citirea datei semnalul RDY trece în 1 logic. Desfăşurarea acestui transfer este prezentată în fig. 2.19.a.

Trebuie remarcat faptul că acest transfer se poate face şi de la un dispozitiv extern ce nu respectă întrutotul protocolul prezentat.

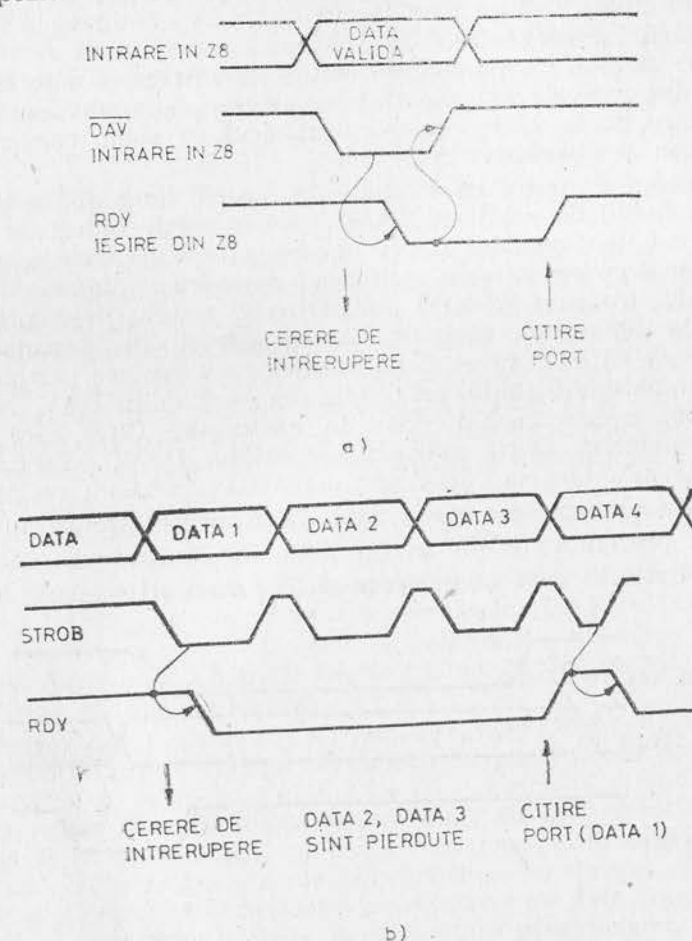


Fig. 2.19. Recepţia datelor paralele cu semnale de control. a) cazul standard b) posibilitatea pierderii datelor

Ne referim la cazul perifericelor care furnizează data însoţită de un strob (un impuls de lăţime determinată). Cu această primă degradare a protocolului transferul poate funcţiona cu condiţia ca dispozitivul extern să nu lanseze un nou transfer pînă cînd Z8-ul nu confirmă că e gata pentru a primi altă dată (RDY = 1). Dacă dispozitivul extern nu ţine seama de semnalul de RDY, atunci datele care sosesc la Z8 atît timp cît RDY e jos nu sînt zăvorîte în registrul de intrare. Prima dată preluată nu va fi pierdută prin suprainscriere, dar datele ulterioare se vor pierde (v. fig. 2. 19b). Această situaţie a fost analizată pentru că poate să apară şi să aducă prejudicii îndeosebi dacă Z8-ul execută concomitent cu transferul şi alte sarcini. Cu precauţiile cerute de situaţie şi datorită faptului că dispozitivele externe sînt în general mult mai lente decît rata la care poate prelua microcalculatorul se poate comunica şi cu acest gen de dispozitive în siguranţă.

În regimul de ieşire cu semnale de control linia P35 e folosită pentru semnalul de validare, DAV0, care însoţeşte datele de ieşire de pe portul P0. Semnalul DAV0 devine activ după scrierea portului P0 numai cu condiţia ca perifericul de ieşire să ţină un 1 logic pe linia de intrare P32(RDY). Dispozitivul extern, după primirea datei de la Z8, trebuie să indice cu un semnal de recunoaştere (RDY = 0) că a acceptat data, această recunoaştere permite încheierea transferului iniţial prin ridicarea semnalului DAV. În acelaşi moment se generează o cerere de întrerupere, IRQ2, care poate fi sau nu folosită. După ridicarea semnalului DAVO, dispozitivul extern este autorizat să ridice semnalul RDY la 1 logic, confirmînd că este gata să accepte o nouă dată. Desfăşurarea transferului de ieşire este prezentată grafic în fig. 2.20. Se menţionează că după înscrierea datei în port, se pot face şi alte înscrieri în port. În re-

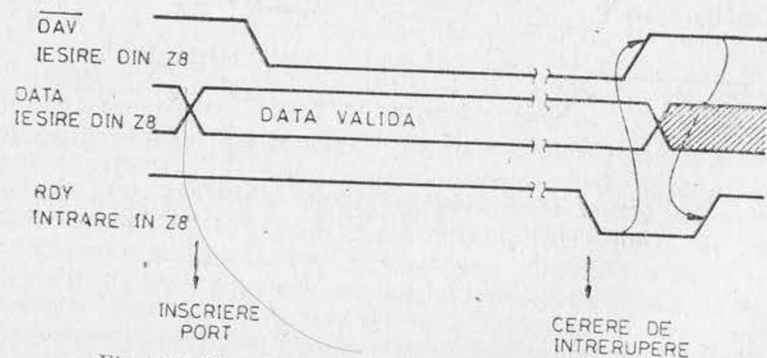


Fig. 2.20. Transferul datelor paralele spre un periferic

gistrul de ieşire rămînînd ultima dată înscrisă. Înscrierea şi supraînscrierea portului constă din specificarea portului P0 ca destinaţie într-o instrucţiune. Pentru a evita supraînscrierea, este utilă aşteptarea cererii de întrerupere IRQ2 sau testarea liniei P32 (RDY) pînă cînd aceasta trece în 0 logic. În cazul în care perifericul acceptă date însoţite de strob se poate, în locul semnalelor de intercondiţionare, RDY şi DAV, să se lege intrarea RDY (P32) la ieşirea DAV (P35).

Linia P32 (P35) reprezintă fie o intrare (ieşire) de uz general fie un semnal de interfaţare a portului P0. Calitatea semnalului de interfaţare, Ready respectiv Data available este dată de sensul de vehiculare a datelor prin port.

2.4.3. PORTUL P1

Portul P1 poate fi configurat fie ca port de intrare sau ieşire cu sau fără semnale de validare şi recunoaştere, fie ca port de adrese şi date utilizat în adresarea memoriei externe sau a unor circuite de intrare/ieşire suplimentare.

Acest port poate fi configurat programînd registrul R248 (PO1M). El nu admite programarea individuală a liniilor şi nici măcar a semiocteţilor. Cînd e folosit ca port de intrare sau ieşire accesul la pinii externi se face prin intermediul registrului dedicat R1. Pentru configurarea portului P1 ca port de intrare (sau ieşire) cu semnale de validare şi recunoaştere, se programează pe lîngă R248 (PO1M) şi registrul R247(P3M).

Semnalele de validare şi recunoaştere apar pe liniile P33 şi P34. Dacă e intrare, pe linia P33 apare semnalul DAV ce însoţeşte datele de la periferic. Acest semnal zăvoreşte datele în registrul de intrare după care Z8 confirmă zăvorîrea datelor perifericului trăgînd linia P33 (RDY) la zero.

Pentru adresarea resurselor externe (memorii şi circuite de intrare/ieşire), portul P1 trebuie configurat pentru a furniza adresele, A0—A7 şi a vehicula datele D0—D7. Relaţiile temporale dintre aceste semnale multiplexate şi semnalele de control însoţitoare AS, DS şi R/W au fost deja prezentate în paragraful precedent. În această configuraţie se poate folosi o linie de control suplimentară dacă se doreşte implementarea de memorie de date. Pentru aceasta există o linie dedicată, dar care necesită programarea registrului P3M (D3 D4 — 10 sau 01). În acest caz linia P34 va furniza semnalul DM (External Data Memory). Numai prin portul P1 se poate

adresa 256 de locații de memorie externă sau 512 dacă se organizează pe lîngă memoria program și memoria de date. În acest mod portul P1 nu mai poate fi accesat ca un registru. Pe de altă parte și acest port poate fi trecut în starea de impedanță ridicată împreună cu liniile de control AS, DS și R/W (împreună și cu P0 dacă acesta e organizat ca port de adrese). Se permite în acest fel ca Z8 să fie utilizat în configurații multiprocesor sau DMA. Dacă considerăm o astfel de configurație multiprocesor se poate dezvolta o logică prin care fiecare procesor din rețea să devină master, prin utilizarea a două semnale de control suplimentare P33 (intrare) și P34 (ieșire). Pe linia P34, asignată ca cerere de magistrale — BUSRQ, Z8 poate cere magistrala resurselor comune, cedarea magistralei comune de către un alt microcalculator Z8 este semnalizată pe linia de intrare, P33, asignată ca intrare de recunoaștere—BUSACK. În timp ce pentru cel ce cere, asignările sînt: P34 (BUSRQ) și P33(BUSACK), pentru cel care cedează, asignările trebuie să fie inverse: P34(BUSACK) și P33(BUSRQ). Cedarea magistralei comune altor microcalculatoare cît și confirmarea primirii magistralei se fac sub controlul programului prin tratarea întreruperii externe IRQ1 (P33). Prin magistrala resurselor comune înțelegem semnalele AD0—AD7, A8—A15, AS, DS și R/W.

2.4.4. PORTUL P2

Portul se poate configura linie cu linie ca intrare sau ieșire prin registru R246(P2M). Portul poate fi accesat prin registrul R2, asemănător cu cele prezentate la portul P0 și P1. Portul P2 poate lucra și ca port de intrare/ieșire cu semnale de validare și recunoaștere; și întrucît aceste linii provin tot din portul P3, alegerea acestui mod se face tot prin registrul R247 (P3M). Pentru aceste semnale de control sînt afectate liniile P31 și P36. Aceste linii devin pe rînd DAV sau RDY respectiv RDY sau DAV funcție de direcția intrare/ieșire a bitului 7 din port. Portul P2 este disponibil întotdeauna pentru operații de intrare/ieșire. Atunci cînd e programat (fie și o singură linie) ca ieșire, liniile pot fi configurate ca ieșiri cu drena în gol, prin bitul D0 din registrul P3M.

2.4.5. PORTUL P3

Portul P3 poate fi configurat ca port de intrări/ieșiri sau ca port de control. La funcțiile sale de control s-au mai făcut referiri cu ocazia prezentării celorlalte trei porturi. Indiferent de modul de

configurare, liniile P30—P33 sînt totdeauna intrări iar liniile P34—P37 sînt întotdeauna ieșiri.

Citirea și scrierea portului P3 se face prin registrul dedicat R3. Citind registrul R3, citim valorile prezente pe cele 4 linii de intrare P30—P33 și data din registrul de ieșire anterior scrisă (v. fig. 2.18.). Scrierea celor 4 biți ai registrului de ieșire e posibilă numai în cazul că sînt configurați ca ieșiri generale. Funcțiile pe care le poate îndeplini portul P3 se programează prin registrul de control și configurare R247(P3M). Funcțiile posibile ale portului P3 se materializează prin următoarele semnale de control:

— semnale de validare și recunoaștere pentru porturile P0, P1 și P2 (DAV și RDY),
— semnale de cerere de întrerupere (IRQ0—IRQ3),
— intrare și ieșire din circuitul de serializare/deserializare (SI și S0) și
— selecția bancului de memorie externă de date (DM).

Evident că ele nu pot fi prezente simultan în funcționare.

În tabelul următor se prezintă aceste semnale și pinii din portul P3.

Cele patru linii de intrare, P30—P33, indiferent de configurații, pot întrerupe microcalculatorul dacă se permite această întrerupere.

Tabelul nr. 2.2.

Linia	Semnalul	Observații
P31 P32 P33 P34 P35 P36	DAV2/RDY2 DAV0/RDY0 DAV1/RDY1 RDY1/DAV1 RDY0/DAV0 RDY2/DAV2	Semnale de control interfață
P30 P31	IRQ3 IRQ2	Cereri de întrerupere
P32 P33	IRQ0 IRQ1	Cereri de întrerupere
P31 P36	TIN TOUT	Intrare/ieșire de la consolă
P34	DM	Selecție memorie de date
P30 P37	SI S0	Intrare/ieșire serie

2.5. CIRCUITELE DE CEAS

Microcalculatorul Z8 are două circuite de ceas. Fiecare circuit de ceas este compus dintr-un predivizor de 6 biţi şi un numărător programabil de 8 biţi.

În fig. 2.21 este prezentată schema bloc a circuitelor de ceas. Ce nu se observă din diagramă este faptul că predivizorul ceasului T0 este atacat numai de tactul intern pe cînd predivizorul ceasului T1 poate fi atacat atît cu tact intern cît şi cu tact extern. Prezenţa acestor circuite de ceas sporeşte valoarea microcalculatorului integrat în special în aplicaţiile în timp real. Numărătoarele odată pornite lucrează independent de programul în desfăşurare a microcalculatorului, legătura cu programul procesorului putîndu-se face în momente mai puţin critice de timp şi bineînţeles prin sistemul de întreruperi.

Numărătoarele pot fi lansate, oprite, repornite programînd respectiv reprogramînd registrul dedicat R241(TMR). Repornirea numărătoarelor se poate face continuînd numărarea (divizarea) din momentul opririi sau reluînd divizarea de la valoarea iniţială înscrisă în numărător.

Numărătoarele pot fi de asemenea programate să se oprească automat la atingerea valorii zero (un singur pas) sau să se reîncarce cu valoarea iniţială la atingerea valorii zero şi să continue numărarea înapoi pînă la atingerea din nou a valorii zero ş.a.m.d. (numărare continuă sau altfel spus numărare modulo n). Programarea acestor două moduri posibile de utilizare a numărătoarelor se face prin bitul D0 al registrelor dedicate PRE0 şi PRE1. Tot prin registrele PRE0 şi PRE1 se programează valoarea (6 biţi) prin care predivizorul împarte frecvenţa tactului de intrare. Valoarea înscrisă în predivizor poate fi orice valoare între 0 şi 63 ce corespunde la o divizare cu 64, 1, 2 pînă la 63. Semnalul de la ieşirea din fiecare predivizor atacă numărătorul pereche, care-şi decrementează valoarea înscrisă în registrul numărătorului. Cînd numărătorul îşi atinge valoarea zero se generează în mod automat o cerere de întrerupere, IRQ4 (corespunzător la T0) sau IRQ5 (corespunzător la T1). Cînd întreruperea nu e dorită se maschează cererea de întrerupere în cauză prin registrul dedicat IMR (vezi paragraful 2.8.). Din schema bloc a circuitelor de ceas prezentată se poate observa că numai numărătorul se poate citi. Datorită registrului de citire a numărătorului (COUNTER READOUT REGISTER) numărătorul poate fi citit oricînd în timpul numărării fără a deranja procesul de numărare în desfăşurare.

Frecvenţa maximă la care poate lucra predivizorul şi numărătorul este de 1 MHz. Numărătorul T0 şi predivizorul (PRE0) poate

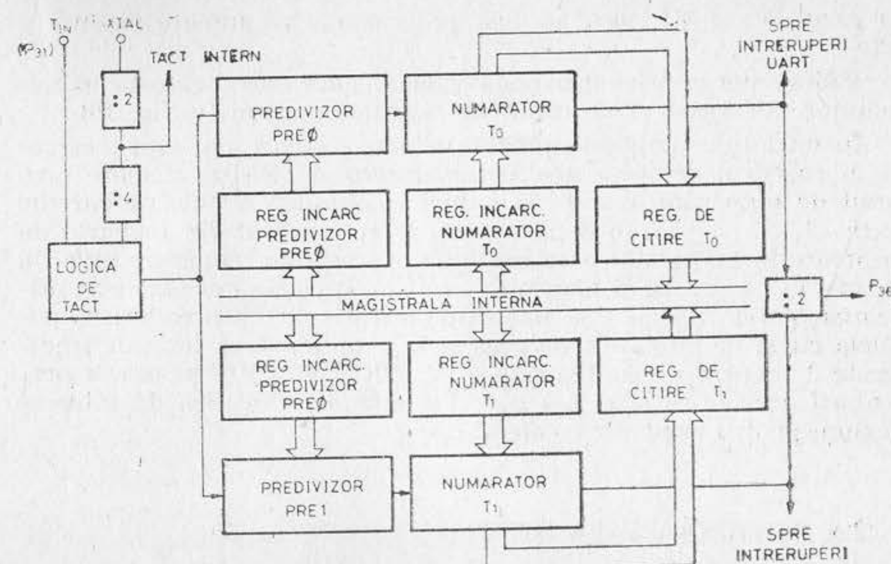


Fig. 2.21. Diagrama circuitelor de ceas

fi atacat numai de tactul intern (4MHz) divizat la rîndul său cu 4. Tactul intern fiind divizat cu 2 faţă de frecvenţa cristalului de cuarţ, care e maxim 8MHz. Numărătorul (T1) şi predivizorul său (PRE1) poate fi atacat atît de tactul intern cît şi de un tact extern. În cazul că numărătorul T1 numără pe tactul intern, intrarea TIN adică P31, se poate folosi pentru startarea, sincronizarea şi repornirea numărării cu un semnal extern. Aceste moduri, ce se referă numai la numărătorul T1, se programează şi se reprogramează prin registrul R241 (TMR).

Datorită faptului că pentru circuitele de ceas sînt afectaţi numai doi pini (TIN—P31, TOUT—P36), legarea în cascadă a celor două numărătoare se poate face numai în maniera următoare:

1. ieşirea din T0 este scoasă pe TOUT prin programarea registrului TMR (intrarea lui T0 fiind totdeauna pe tactul intern),

2. TOUT se leagă la TIN care atacă numărătorul T1,

3. se autorizează întreruperea IRQ5, întrucît T1 nu are la dispoziţie alt pin de ieşire, funcţionarea cascadei T0—T1 se face numai prin întrerupere.

Prin pinul P36 se poate scoate în afară fie T0OUT/2, fie T1OUT/2, fie tactul intern/2. Factorul de umplere în cazul tactului intern şi

în cazul T0 şi T1 dacă au fost programate să numere modulo n este 1/2.

P36 se şterge oricînd o nouă valoare „m" este încărcată în numărător din registrul de încărcare a numărătorului (v. fig. 19).

În modul de numărare modulo n noua valoare, m, este încărcată în registrul de încărcare a numărătorului fără să afecteze procesul de numărare în curs şi numai la sfîrşitul numărării curente (n, n—1, ...1, 0) noua valoare trece automat din registrul de încărcare în numărător care continuă operaţia de numărare inversă, m, m—1, ... ca şi cum nu s-ar fi întîmplat nimic. Întrucît utilizarea pinilor P31 şi P36 sînt disputaţi atît de intrarea/ieşirea paralelă cît şi de circuitele de ceas şi în configurarea lor sînt amestecate 3 registre de configurare P3M, PRE1 şi TMR cu ocazia prezentării acestor registre s-a dat şi o schemă simbolică de interconexiuni pentru acest caz (v. fig. 2.26).

2.6. INTRAREA/IEŞIREA SERIE

Microcalculatorul integrat Z8 are în componenta sa un circuit de deserializare pentru recepţie şi un circuit de serializare pentru transmitere.

Atît partea de recepţie cît şi partea de transmitere conţin fiecare cîte un circuit de verificare a parităţii, respectiv de generare a parităţii. Registrul R240 (SIO) care serveşte partea de intrare/ieşire serie este de fapt format din două registre, unul de intrare şi celălalt de ieşire avînd aceeaşi adresă. Schema bloc a circuitului de intrare/ieşire serie este prezentată în fig. 2.22. Tot în această schemă bloc a fost inclusă şi partea din circuitul de ceas T0 care este destinată formării tactului de recepţie — transmitere serie. În acest sens T0 are o serie de legături care sînt dedicate. Din portul P3 liniile P30 şi P37 sînt dedicate intrării serie respectiv ieşirii serie.

Pe această structură este implementată legătura serie full duplex asincrona care poate să meargă la o viteză maximă de 62500 biţi/sec. Această rată de transfer provine din faptul că frecvenţa maximă în Numărător (T0) este de 1MHz iar tactul necesar circuitelor de serializare/deserializare e de 16 ori mai rapid decît tactul efectiv de deplasare (1 Mhz 16 = 62500).

Data de transmis este încărcată în registrul R240 şi deplasat afară bit cu bit prin linia P37 după ce în faţa bitului D0 s-a inserat un bit de start (0) urmînd ca după ultimul bit de date să se însereze unul sau doi biţi de stop (1). Se transmit totdeauna 8 biţi

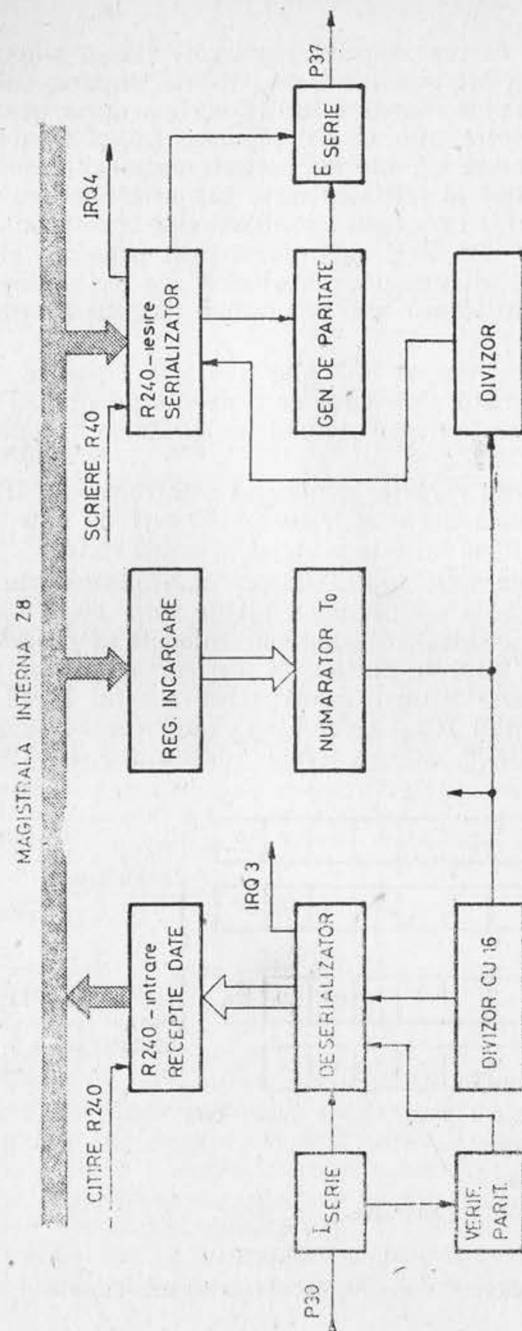


Fig. 2.22. Diagrama circuitului de intrare/ieşire serie

indiferent dacă s-a optat pentru paritate. Dacă s-a optat pentru paritate, al 8-lea bit este bitul de paritate impară, data transmisă fiind în acest caz de numai 7 biți. Dacă s-a optat pentru transmisia cu paritate între ultimul bit de date, D6, și bitul sau biții de stop, se intercalează un bit de paritate impară. Recepția serie se face prin linia P30 și este deplasată bit cu bit în registrul DESERIALIZATOR după care data asamblată este transferată în registrul de intrare R240. De aici, utilizatorul prin program citește valoarea recepționată. În timpul deserializării are loc și procesul de verificare a parității (dacă s-a programat recepția/transmisia cu bit de paritate).

Între caractere, ieșirea P37 este ținută la 1 pentru a putea face discriminarea următorului caracter transmis pe linie. Formatul datelor transmise și formatul datelor așteptate la recepție este prezentat în fig. 2.23.

Transmisia unui caracter generează o intrerupere, IRQ4, de care trebuie ținut seamă dacă se transmit fluxuri de date, întrucît registrul de ieșire R240 nu este protejat la supra scriere.

La recepție, data trebuie să fie prefațată obligatoriu de bitul de start și postfațată de cel puțin un bit de stop. Dacă s-a optat pentru paritate atunci bitul al 8-lea este înlocuit în procesul de serializare cu 0 dacă bitul de paritate a fost corespunzător și cu 1 dacă a apărut o eroare. Pentru valorificarea acestui lucru utilizatorul la citirea registrului R240 trebuie să examineze acest bit. O cerere la întrerupere, IRQ3, este generată întotdeauna după deserializare

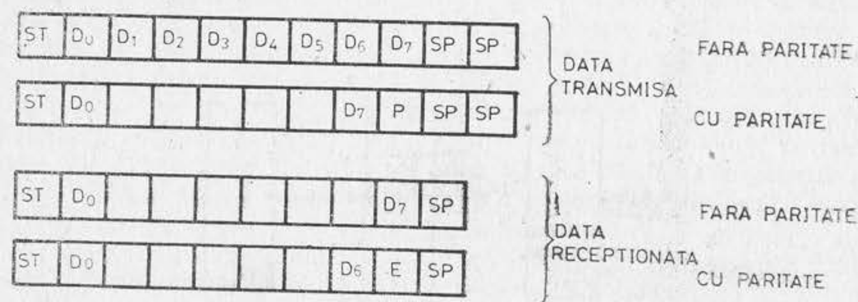


Fig. 2.23. Formatul datelor pe transmisia serie

la transferul informației în registrul R240. Este la latitudinea utilizatorului utilizarea sau nu a acestei întreruperi. De asemenea trebuie reținut faptul că dacă utilizatorul nu preia la timp caracterul recepționat, următorul caracter se suprapune pe acesta nerămînînd nici o indicație asupra faptului că s-a pierdut un caracter sau mai multe; deci utilizarea întreruperii IRQ3(S1) este aproape obligatorie.

2.7. ÎNTRERUPERILE

Microcalculatorul integrat Z8 mînuiește la un moment dat 6 întreruperi diferite. Întreruperile pot fi mascate, aranjate într-o ordine de prioritate dorită prin R251(IMR), respectiv R249(IPR). Întreruperile pot fi mascate individual sau dezactivate în bloc prin bitul D7 din registrul R251. Întreruperile pot fi culese din 8 locuri, conform cu tabelul următor:

NUME	LOCAȚIA VECT.INT.	SURSA	COMENTARIU
IRQ0	0000H	DAV0, Intrare P32	P32—extern, front negativ
IRQ1	0002H	DAV1, Intrare P33	P33—extern, front negativ
IRQ2	0004H	DAV2, Intrare P31, TIN	P31—extern, front negativ
IRQ3	0006H	Intrare P30 SI	P30—extern, front negativ intern
IRQ4	0008H	T0 SO	intern intern
IRQ5	000AH	T1	intern

Cînd o cerere de întrerupere e nemascată și are prioritate, microcalculatorul Z8 trece controlul subrutinei de tratare a evenimentului preluînd din locația dedicată adresa acesteia. Aceasta presupune că la scrierea memoriei program locațiile vectorilor de întrerupere au fost înscrise cu adresele corespunzătoare subrutinelor de întrerupere. Deoarece numărătorul T0 este destinat circuitului de I/E serie întreruperea generată de T0 și de circuitul de ieșire serie SO, pot împărți aceeași cerere de întrerupere (IRQ4). Pe de

altă parte, întreruperea pe linia de intrare P30 (front negativ) împarte aceeași cerere de întrerupere, IRQ3, cu întreruperea generată de circuitul de recepție serie (SI). Și aceasta este posibil fără nici o pierdere întrucît linia P30 este folosită ca linie de intrare serie.

Modificarea registrului de mascare a întreruperilor, R251, sau a registrului de stabilire a priorităților, R249, trebuie făcută în condițiile în care s-au dezactivat global întreruperile prin instrucțiunea DI sau prin altă instrucțiune prin care se șterge bitul D7 din registrul R251. În figura 2.24 este prezentată schema bloc a circuitului de tratare a întreruperilor.

Cînd o cerere de întrerupere e acceptată, Z8 intră într-un ciclu special de tratare a întreruperilor care:

1) dezactivează global întreruperile,

2) salvează în stivă contorul program și indicatorii de stare,

3) șterge bitul cererii de întrerupere servită din R250 (IRQ),

4) preia adresa din locația vectorului de întrerupere în contorul program, și

5) începe rularea subrutinei.

Cererile de întrerupere sînt testate înainte de fiecare ciclu de aducere a unei noi instrucțiuni. Cererile externe (P30—P33) sînt testate cu 4 perioade de tact înaintea începerii ciclului de aducere a instrucțiunii următoare. Cererile de întreruperi interne (S1, S0, T1, T0) sînt testate cu numai o perioadă de tact înainte.

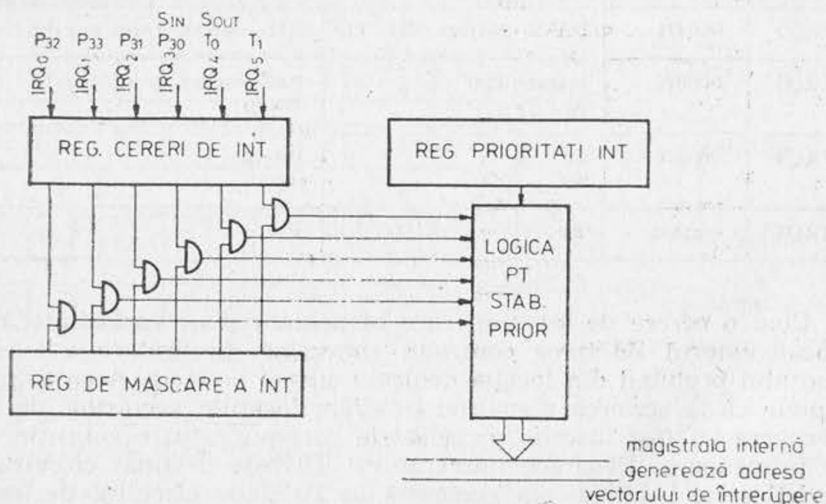


Fig. 2.24. Diagrama circuitului de tratare a întreruperilor

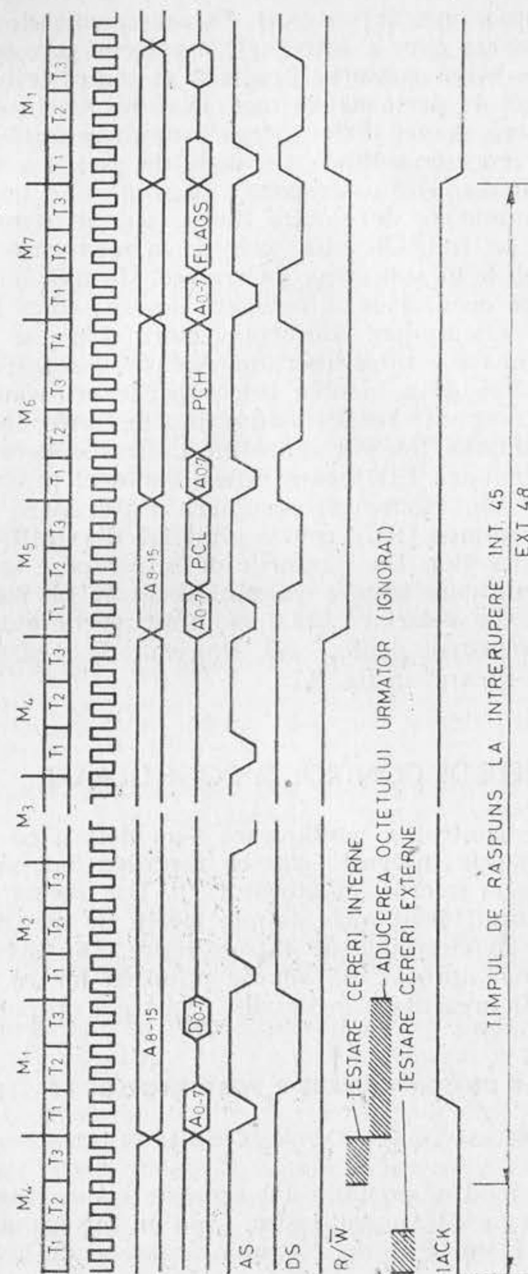


Fig. 2.25. Formele de undă asociate tratării unei întreruperi

Pentru tratarea unei întreruperi Z8 are nevoie de 7 cicli mașină pentru a vedea cine a întrerupt, a selecta vectorul de întrerupere corect, a salva contorul program și indicatorii de stare. Aceasta înseamnă 44 perioade de tact. Dacă se iau în calcul și perioadele de testare atunci întreruperea internă cere 45 de perioade iar cea externă consumă 48 de perioade de tact. La un cuarț de 8MHz asta înseamnă 11,25 us respectiv 12µs.

Numai variantele de dezvoltare Z8/64, pun în exterior semnalul IACK, activ pe 1; IACK este activ de la recunoașterea întreruperii pînă la saltul la subrutina de tratare efectivă a întreruperii.

La terminarea ciclului de întrerupere descris, adică la ajungerea în subrutina de întrerupere conform vectorului nu se reface bitul de autorizare globală a întreruperilor. Aceasta înseamnă că pentru a avea posibilitatea de a încuiba întreruperile, utilizatorul trebuie să insereze la începutul fiecărei subrutine de întrerupere instrucțiunea EI. Terminarea fiecărei subrutine de întrerupere trebuie făcută prin instrucțiunea IRET care reface contorul program și indicatorii de stare din momentul acceptării întreruperii servite. De asemenea instrucțiunea IRET reface bitul D7 din IMR, de validare globală a întreruperilor. Dacă cererile de întrerupere se maschează, ele pot fi în continuare testate examinînd bit cu bit registrul R250 (IRQ) sau dacă ne referim numai la întreruperile externe, P30—P33, testînd registrul dedicat R3. Procesul de desfășurare a întreruperilor este ilustrat în fig. 2.25.

2.8. REGISTRELE DE CONTROL ȘI CONFIGURARE

Registrele de control și configurare s-au definit ca o parte din memoria citește/scrie integrată sau ca reprezentînd un subset din setul de registre al microcalculatorului Z8. Dar aceste registre dedicate sînt mai mult decît niște simple locații de memorie prin legăturile directe cu circuitele de intrare/ieșire. Din aceste considerente, e mult mai aproape de adevăr numirea lor ca registre. În continuare vor fi prezentate în detaliu registrele grupate din punct de vedere funcțional.

2.8.1. REGISTRELE DE CONFIGURARE A PORTURILOR DE I/E

2.8.1.1. *R246 – REGISTRUL DE MOD AL PORTULUI P2 (P2M)*

Registrul de mod al portului P2 servește la programarea fiecărei linii din port ca intrare sau ieșire. Cînd un bit din acest registru este înscris cu 1 sau 0, linia corespunzătoare e definită ca linie

de intrare respectiv linie de ieșire. După inițializare, în acest registru se găsește valoarea OFFH, astfel încît de la punerea sub tensiune pînă la configurarea dorită de utilizator liniile portului P2 nu deranjează interfața. Specificarea tipului de ieșire pentru liniile definite ca ieșire și precizarea dacă P2 va fi însoțit de semnale de validare și recunoaștere, se face prin registrul de mod al portului P3. Registrul R246 poate fi numai încărcat, el nu poate fi citit.

2.8.1.2. *R247 – REGISTRUL DE MOD AL PORTULUI P3 (P3M)*

Reamintim că liniile portului P3 sînt fixate, în ceea ce privește sensul (P30—P33 sînt intrări iar P34—P37 sînt ieșiri), astfel că stabilirea sensului nu face obiectul programării. Prin P3M se specifică modul de alocare a liniilor multifuncționale (intrări/ieșiri, intrări/ieșiri serie, cereri de întrerupere, intrare/ieșire de ceas semnale de validare și recunoaștere, ieșire de stare). Acest registru poate fi numai scris.

PARITATEA (D7). Dacă s-a optat pentru utilizarea circuitului de intrare/ieșire serie (prin bitul D6) înscriind bitul D7 atunci:

1. în transmisii, data emisă are automat paritatea impară:

BIT PARIT.	D6	D5	D4	D3	D2	D1	D0

2. la recepție se calculează paritatea impară pe primii 7 biți și se compară cu bitul de paritate din caracterul recepționat.

LINIILE P30 și P37 (D6). Dacă D6 = 1 linia P30 se conectează la intrarea circuitului de recepție serie iar linia P37 este conectată la ieșirea circuitului de ieșire serie; altfel, D6 = 0, liniile P30 și P37 sînt liniile de intrare respectiv de ieșire de uz general.

LINIILE P31 și P36 (D5). Dacă D5 = 0, liniile P31 și P36 sînt linii de intrare respectiv ieșire, de uz general sau, în colaborare cu D1 din registrul predivizor PRE0, P31 devine intrare în registrul T1(TIN), iar P36 devine ieșire fie pentru numărătoarele T0 sau T1 fie direct pentru tactul intern funcție de valoarea biților D6, D7 din registrul R241 (TMR). Trebuie remarcat faptul că D5 = 0 din acest registru, P3M, nu este suficient pentru a defini linia P36 de ieșire de uz general; pentru aceasta se impune ca prin R241 (TMR) linia să nu fie alocată circuitelor de ceas, adică D6, D7 din acest registru

să fie 00. În fig. 2.26 se prezintă o schemă simbolică a tuturor conexiunilor interne care afectează pinii P31 și P36. Liniile P31 și P36 sînt folosite și ca linii de validare și recunoaștere pentru portul P2 dacă D5 este înscris. Dacă liniile P31 și P36 sînt alese ca linii de validare și recunoaștere pentru portul P2, biții D6 și D7 din TMR nu mai au nici o influență.

LINIILE P33 și P34 (D4, D3). Funcție de valoarea biților D4, D3 liniile sînt:

— intrare/ieșire de uz general (00),

— intrare/DM (01 sau 10) și

— linii de validare și recunoaștere (DAV1/RDY1 respectiv RDY1/DAV1 — 11).

LINIILE P32 și P35 (D2). Prin bitul D2 se specifică dacă liniile P32 și P35 sînt linii de intrare respectiv ieșire, de uz general, sau linii de validare și recunoaștere pentru portul P0(DAV0/RDY0 și RDY0/DAV0).

ETAJELE DE IEȘIRE ALE LINIILOR PORTULUI P2 (D0). Dacă D0 = 1, înseamnă că ieșirile portului P2 sînt identice cu cele ale portului P0 și P1 și sînt compatibile TTL. D0 = 0 înseamnă că liniile de ieșire ale portului P2 au drena în gol (open drain). Ieșirile cu drena în gol se pot lega în SAU cablat cu alte ieșiri de același tip (asemănător cu ieșirea cunoscută la circuitele TTL de colector în gol).

2.8.1.3. *R248 (P01M) – REGISTRUL DE MOD AL PORTURILOR P0 și P1*

Prin acest registru se configurează utilizarea porturilor P0 și P1, se alege stiva în spațiul intern sau extern de memorie și se alege un timp prelungit de acces la memoria externă.

PORTUL P0. (D0, D1 și D7, D6). Portul P0 poate fi configurat la nivel de semiocteți. La punerea în funcțiune (inițializare) acest port se găsește configurat ca intrare. Biții D7, D6 configurează semioctetul superior, iar în cazul în care este configurat ca linii de adresă, A12—A15, atunci după cum e și logic, aceasta are influență și asupra semioctetului inferior care devine automat A8—A11, independent de programarea biților D0, D1 responsabili de acest semioctet.

D7	D6	P04 — P07
0	0	— ieșiri
0	1	— intrări
1	x	— linii de adresă A12—A15

D7	D1	D0	P00 — P03
0	0	0	— ieșiri
0	0	1	— intrări
0	1	x	— linii de adresă A8—A11
1	x	x	— linii de adresă A8—A11

PORTUL P1 (D3, D4). Biții D3, D4 sînt destinați configurării portului P1. Și acest port după inițializare se găsește în modul intrare.

D4	D3	P10 — P17
0	0	— ieșiri
0	1	— intrări
1	0	— adrese/date multiplexate
1	1	— impedanță ridicată

Starea de impedanță ridicată programată pentru portul P1 se răsfrînge și asupra portului P0, parțial sau total, după cum e folosit pentru accesul la memoria externă. Această interdependență care cuprinde pe lingă biții D4 D3 și D7 și D1 este prezentată mai jos:

D7	D4	D3	D1	impedanța ridicată
0	1	1	0	— P1, AS, DS, și R/W
0	1	1	1	— P1, P00—P03, AS, DS și R/W
1	1	1	X	— P1, P0, AS, DS și R/W

EXTINDEREA CICLULUI DE MEMORIE. Prin D5=0, ciclul de memorie se extinde cu o perioadă pentru a permite interfațarea cu memorii mai puțin rapide. În mod normal (D5=1), se pot interfața memorii cu ciclu de acces mai mic de 500 ns; prin extinderea ciclului, se pot interfața memorii cu ciclu mai mic de 750 ns.

LOCALIZAREA STIVEI. Cînd D2 este 1, stiva este internă, adică în setul de registre, de uz general R127 — R4. Fixarea bazei stivei se face prin registrul R255 (SPL). Dacă se alege lucru cu stiva în memoria externă, D2=0, atunci fixarea stivei este dată de registrele R254 și R255 (SPH, SPL). După inițializare, stiva este localizată în setul de registre de uz general și baza stivei este indicată de valoarea din registrul R254 (SPL); și întrucît valoarea de trezire a acestui registru este nedefinită, după inițializare urmează obligatoriu înscrierea acestui registru. Și acest registru poate fi numai înscris.

2.8.2. STIVA, INDICATORI DE STARE, POINTERUL DE REGISTRE

2.8.2.1. *INDICATORUL STIVEI R255, R254*

Așa cum am arătat deja, dacă stiva este internă, atunci este suficient registrul R255 (SPL) pentru indicarea poziției curente a stivei. În acest caz, registrul dedicat R254 (SPH) rămîne la dispoziția utilizatorului ca registru de uz general.

2.8.2.2. *R253 INDICATORUL (POINTERUL) DE REGISTRE (RP)*

Numai semioctetul superior al acestui registru este folosit în mecanismul de adresare pe 4 biți a registrelor interne, mecanism descris deja. În semioctetul inferior se citește totdeauna 0, iar în momentul scrierii, valoarea lui nu are importanță.

V7	V6	V5	V4	X	X	X	X	RP

2.8.2.3. *R252. INDICATORII DE STARE*

Biții D2—D7 conțin indicatorii de stare; D1 și D0 pot fi utilizați în scopuri generale de către programator, de aceea se mai numesc și indicatori utilizator (USER FLAGS).

D7	D6	D5	D4	D3	D2	D1	D0

F2 F1 – INDICATORI UTIL.
H – TRANSPORT LA JUMĂTATE
D – AJUSTARE ZECIMALĂ
V – DEPĂȘIRE
S – BIT DE SEMN
Z – ZERO
C – TRANSPORT

Indicatorii C, Z, S, V pot fi folosiți de programator cu instrucțiunile de salt și salt relativ. În condițiile de salt acești indicatori pot fi folosiți individual sau în legătură cu alții creind un repertoriu de 16 condiții. Indicatorii de stare sînt înscriși sau șterși în urma executării unor instrucțiuni. Instrucțiunile și modul cum afectează indicatorii de stare sînt prezentate în paragraful 2.11.

2.8.3. REGISTRELE DE TRATARE A ÎNTRERUPERILOR

2.8.3.1. *R249 REGISTRUL DE PRIORITĂȚI A ÎNTRERUPERILOR (IPR)*

Și acest registru este numai inscriptibil. Acest registru stabilește ordinea de prioritate între cele 6 surse de întreruperi mînuite odată. Aceste întreruperi pot fi aranjate în 48 de moduri diferite. Cele 6 niveluri de întreruperi sînt împărțite în 3 grupe; fiecare grupă este formată din 2 cereri de întrerupere. Grupul A este format din IRQ3 și IRQ5, grupul B din IRQ0 și IRQ2, iar grupul C din IRQ1 și IRQ4. Biții D1, D2 și D5 stabilesc prioritatea în interiorul grupului.

Grup C	D1 = 0	IRQ1 > IRQ4
	D1 = 1	IRQ1 < IRQ4
Grup B	D2 = 0	IRQ2 > IRQ0
	D2 = 1	IRQ2 < IRQ0
Grup A	D5 = 0	IRQ5 > IRQ3
	D5 = 1	IRQ5 < IRQ3

Prioritatea între grupuri se stabilește prin biții D0, D3 și D4 pe 6 combinații după cum urmează:

D4	D3	D0	Prioritate mare > medie > mică		
0	0	0		nefolosit	
0	0	1	C	A	B
0	1	0	A	B	C
0	1	1	A	C	B
1	0	0	B	C	A
1	0	1	C	B	A
1	1	0	B	A	C
1	1	1		nefolosit	

Biții D6 și D7 nu sînt folosiți. Pentru a ușura alegerea priorităților dorite a celor 6 întreruperi mînuite odată, în tabelul nr. 2.3 se prezintă *in extenso* codurile celor 48 de aranjamente posibile.

Tabelul nr. 2.3. CODURILE COMBINAȚIILOR POSIBILE DE PRIORITĂȚI

IRQ1>IRQ4>IRQ5>IRQ3>IRQ2>IRQ0 01	IRQ1>IRQ4>IRQ3>IRQ5>IRQ2>IRQ0 21
IRQ4>IRQ1>IRQ5>IRQ3>IRQ2>IRQ0 03	IRQ4>IRQ1>IRQ3>IRQ5>IRQ2>IRQ0 23
IRQ1>IRQ4>IRQ5>IRQ3>IRQ0>IRQ2 05	IRQ1>IRQ4>IRQ3>IRQ5>IRQ0>IRQ2 25
IRQ4>IRQ1>IRQ5>IRQ3>IRQ0>IRQ2 07	IRQ4>IRQ1>IRQ3>IRQ5>IRQ0>IRQ2 27
IRQ5>IRQ3>IRQ2>IRQ0>IRQ1>IRQ4 08	IRQ3>IRQ5>IRQ2>IRQ0>IRQ1>IRQ4 28
IRQ5>IRQ3>IRQ1>IRQ4>IRQ2>IRQ0 09	IRQ3>IRQ5>IRQ1>IRQ4>IRQ2>IRQ0 29
IRQ5>IRQ3>IRQ2>IRQ0>IRQ4>IRQ1 0A	IRQ3>IRQ5>IRQ2>IRQ0>IRQ4>IRQ1 2A
IRQ5>IRQ3>IRQ4>IRQ1>IRQ2>IRQ0 0B	IRQ3>IRQ5>IRQ4>IRQ1>IRQ2>IRQ0 2B
IRQ5>IRQ3>IRQ0>IRQ2>IRQ1>IRQ4 0C	IRQ3>IRQ5>IRQ0>IRQ2>IRQ1>IRQ4 2C
IRQ5>IRQ3>IRQ1>IRQ4>IRQ0>IRQ2 0D	IRQ3>IRQ5>IRQ1>IRQ4>IRQ0>IRQ2 2D
IRQ5>IRQ3>IRQ0>IRQ2>IRQ4>IRQ1 0E	IRQ3>IRQ5>IRQ0>IRQ2>IRQ4>IRQ1 2E
IRQ5>IRQ3>IRQ4>IRQ1>IRQ0>IRQ2 0F	IRQ3>IRQ5>IRQ4>IRQ1>IRQ0>IRQ2 2F
IRQ2>IRQ0>IRQ1>IRQ4>IRQ5>IRQ3 10	IRQ2>IRQ0>IRQ1>IRQ4>IRQ3>IRQ5 30
IRQ1>IRQ4>IRQ2>IRQ0>IRQ5>IRQ3 11	IRQ1>IRQ4>IRQ2>IRQ0>IRQ3>IRQ5 31
IRQ2>IRQ0>IRQ4>IRQ1>IRQ5>IRQ3 12	IRQ2>IRQ0>IRQ4>IRQ1>IRQ3>IRQ5 32
IRQ4>IRQ1>IRQ2>IRQ0>IRQ5>IRQ3 13	IRQ4>IRQ1>IRQ2>IRQ0>IRQ3>IRQ5 33
IRQ0>IRQ2>IRQ1>IRQ4>IRQ5>IRQ3 14	IRQ0>IRQ2>IRQ1>IRQ4>IRQ3>IRQ5 34

Tabelul nr. 2.3. (continuare)

IRQ1>IRQ4>IRQ0>IRQ2>IRQ5>IRQ3 15	IRQ1>IRQ4>IRQ0>IRQ2>IRQ3>IRQ5 35
IRQ0>IRQ2>IRQ4>IRQ1>IRQ5>IRQ3 16	IRQ0>IRQ2>IRQ4>IRQ1>IRQ3>IRQ5 36
IRQ4>IRQ1>IRQ0>IRQ2>IRQ5>IRQ3 17	IRQ4>IRQ1>IRQ0>IRQ2>IRQ3>IRQ5 37
IRQ2>IRQ0>IRQ5>IRQ3>IRQ1>IRQ4 18	IRQ2>IRQ0>IRQ3>IRQ5>IRQ1>IRQ4 38
IRQ2>IRQ0>IRQ5>IRQ3>IRQ4>IRQ1 1A	IRQ2>IRQ0>IRQ3>IRQ5>IRQ4>IRQ1 3A
IRQ0>IRQ2>IRQ5>IRQ3>IRQ1>IRQ4 1C	IRQ0>IRQ2>IRQ3>IRQ5>IRQ1>IRQ4 3C
IRQ0>IRQ2>IRQ5>IRQ3>IRQ4>IRQ1 1E	IRQ0>IRQ2>IRQ3>IRQ5>IRQ4>IRQ1 3E

2.8.3.2. *R250 REGISTRUL CERERILOR DE INTRERUPERE (IRQ)*

Acest registru păstrează cererile de întrerupere. Înscrierea acestui registru se face automat dar și prin instrucțiuni și, în plus, poate fi citit; prin urmare se pot utiliza întreruperile și într-o variantă de „polling" prin dezactivarea globală a întreruperilor și citirea lor în acest registru de cereri.

Cînd o cerere de întrerupere apare, bitul corespunzător din acest registru este înscris și rămîne așa pînă la declanșarea ciclului de servire a întreruperii.

X	X	IRQ5	IRQ4	IRQ3	IRQ2	IRQ1	IRQ0

2.8.3.3. *R251 REGISTRUL DE MASCARE AL INTRERUPERILOR (IMR)*

Prin acest registru se poate masca individual fiecare cerere de întrerupere sau se pot masca global toate întreruperile. Bitul D7 poate fi înscris prin instrucțiunea EI sau șters prin instrucțiunea DI. Tot D7 este șters automat în timpul acceptării unei întreruperi

şi este înscris după executarea unei instrucţiuni de reîntoarcere din subrutina de întrerupere — IRET.

D7	X	D5	D4	D3	D2	D1	D0

1 autorizează IRQ0
1 autorizează IRQ1
1 autorizează IRQ2
1 autorizează IRQ3
1 autorizează IRQ4
1 autorizează IRQ5
1 autorizează întreruperile

Subliniem că D7 trebuie să fie şters înainte de a opera modificări a registrelor R249 (IPR) şi R251 (IMR).

2.8.4. REGISTRELE DEDICATE CIRCUITELOR DE CEAS

2.8.4.1. *R241 REGISTRUL DE CONFIGURARE A NUMĂRĂTOARELOR (TMR)*

Prin acest registru se aleg modurile de funcţionare a numărătoarelor, se încarcă numărătoarele şi predivizoarele şi se autorizează funcţionarea acestora.

IEŞIREA CIRCUITULUI DE CEAS. După cum se ştie, un singur pin, P36, poate fi afectat pentru această funcţie iar biţii D7 şi D6 programează la ce anume e folosit acest pin de ieşire.

D7	D6	TOUT — P36
0	0	neutilizat (P36 ieşire gen.)
0	1	ieşire T0
1	0	ieşire T1
1	1	ieşire tact intern

MODURILE DE UTILIZARE A SEMNALULUI EXTERN CA INTRARE ÎN NUMĂRĂTORUL T1. Prin D4 şi D5 se definesc modurile în care semnalul extern este folosit la intrarea în numărătorul T1. Înainte de definirea acestor moduri trebuie ca T1—IN să fie definit ca intrare din sursa externă (R243, D1).

D5	D4	TIN Folosit ca:	EXPLICAŢII
0	0	tact extern	TIN – intrare de tact extern pentru T1. În acest caz tactul intern intră direct în predivizor (trebuie să fie < 1 MHz)
0	1	semnal de validare a tactului intern	T1 numără pe tactul intern validat de semnalul de intrare de pe P31 (TIN). Dacă ÎNTRERUPERILE sînt validate, se generează o int. cînd TIN trece din 1 în 0.
1	0	semnal de declanşare	T1 este încărcat şi intră în funcţiune numai după tranziţia 1/0 pe semnalul de declanşare TIN. Următoarele tranziţii nu se mai efectuează decît după terminarea numărării.
1	1	semnal de redeclanşare	Analog ca mai sus, cu deosebirea că la tranziţii repetate valoarea iniţială a lui T1 este reîncărcată şi numărarea începe de la început.

VALIDAREA NUMĂRĂTORULUI T1. D3 înscris validează operarea numărătorului T1 împreună cu a predivizorului PRE1. Cînd D3 se şterge, se opreşte numărarea.

ÎNCĂRCAREA NUMĂRĂTORULUI T1. Conţinutul registrului de încărcare T1 şi conţinutul registrului de încărcare PRE1 sînt transferate în numărătorul T1 respectiv predivizorul PRE1 după o perioadă de tact de la înscrierea bitului D2. Este permisă validarea şi încărcarea numărătorului deodată. De asemenea, după încărcare, acest bit se şterge automat.

VALIDAREA NUMĂRĂTORULUI T0. Pornirea numărării se face înscriind bitul D1 în acelaşi mod cu cele prezentate pentru T1.

ÎNCĂRCAREA NUMĂRĂTORULUI T0. Încărcarea numărătorului şi a predivizorului din ceasul T0 se face pe bitul D0 înscris, aşa cum s-a prezentat şi pentru numărătorul T1.

2.8.4.2. *R242 – REGISTRUL NUMĂRĂTORULUI (T1)*

La această adresă F2H(242Z) sînt de fapt două registre: registrul de încarcare şi registrul de citire ale numărătorului T1. Prin acest ultim registru se poate citi valoarea curentă a numărătorului fără a deranja funcţionarea acestuia. R242 se poate scrie cu valori cu-

prinse între (00—FFH), dar trebuie specificat că valoarea 00 are semnificaţia 100H (256). Valoarea din acest registru (de încărcare) e transferată în numărător după poziţionarea bitului D2 din R241 (TMR); dacă şi bitul de validare a numărării este poziţionat, începe numărarea inversă pînă la terminarea acesteia. Terminarea numărării înseamnă atingerea valorii 0 în numărător.

2.8.4.3. *R243 – REGISTRUL PREDIVIZORULUI (PRE1)*

Acest registru se referă la ceasul T1 şi păstrează valoarea iniţială a predivizorului (6 biţi) şi îi corespunde în partea de hard, prezentată în paragraful 2.5. (registrul de încărcare a predivizorului). Tot prin acest registru se defineşte sursa de tact şi modul de numărare.

VALOAREA PREDIVIZORULUI (D2—D7). Aceşti biţi, D2—D7, sînt destinaţi încărcării repetate a predivizorului la atingerea valorii 0. Predivizorul, spre deosebire de numărător, numără tot timpul în regim modulo n.

SURSA DE TACT (D1). Ceasul T1 poate merge de la tactul intern (D1 = 0) sau de la un tact extern preluat prin TIN (P31). Cînd se optează pentru tact extern atunci modurile de utilizare a intrării TIN se codifică în registrul R241(TMR).

MODUL DE NUMĂRARE (D0). Ceasul T1 poate număra modulo — n (continuu) sau o singură dată. Dacă D0 este înscris, numărătorul T1 numără o singură dată în jos, pînă la atingerea valorii 0. La atingerea valorii 0, se emite o cerere de întrerupere, care rămîne la latitudinea programatorului de a o fructifica. Numărarea se poate relua numai la reîncărcarea numărătorului T1 prin comanda corespunzătoare dată prin TMR (R241).

Cînd D0 este şters, numărătorul T1 va număra continu. După încărcarea valorii din registrul de încărcare a numărătorului în numărătorul propriu-zis T1, începe numărarea inversă pînă la atingerea valorii 0, cînd valoarea iniţială din registrul de încărcare se reîncarcă după care se continuă numărarea ş.a.m.d. Acest proces poate fi oprit numai prin ştergerea bitului de validare (D3) din registrul R241 (TMR). De asemenea se poate schimba valoarea din registrul de încărcare, fără să afecteze numărarea în ceas; la atingerea valorii 0 în numărător se încarcă noua valoare de unde începe numărarea inversă.

2.8.4.4. *R244 – REGISTRUL NUMĂRĂTORULUI T0*

Acest registru, dublu de fapt, are aceleaşi funcţii pentru numărătorul T0 ca registrul R242 pentru T1.

2.8.4.5. *R245 – REGISTRUL PREDIVIZORULUI PRE0*

Şi acest registru are aceleaşi funcţii cu cele descrise deja pentru R243 cu deosebirea că serveşte ceasul T0; şi întrucît ceasul T0 are ca sursă numai tactul intern, bitul D1 nu mai are nici o funcţie. S-a observat că există anumite interdependenţe între registrele P3M, TRM şi PRE1 în ceea ce priveşte utilizarea liniilor P31 şi P36. Pentru a clarifica aspectele legate de utilizarea acestor linii în fig. 2.26 se prezintă o schemă simbolică a conexiunilor interne controlate de registrele amintite.

2.8.5. **R240 – REGISTRUL DE INTRARE/IEŞIRE SERIE (SIO)**

Utilizarea acestui registru presupune că anterior s-au programat liniile P30 şi P37 ca intrări respectiv ieşiri serie (D6 = 1 din R247). Presupune de asemenea utilizarea ceasului T0 pentru stabilirea vitezei de comunicaţii. Acest registru este format de fapt din două, aşa cum a rezultat şi din prezentarea circuitului de intrare/ieşire serie. Dacă s-a selectat transmisia cu paritate, atunci aceasta e totdeauna impară; în transmisie în acest caz bitul D7 este bitul de paritate care se calculează pe primii 7 biţi şi se inserează automat; la recepţie bitul D7 este un bit de eroare care rezultă din verificarea parităţii caracterului recepţionat. Bitul D7 din caracterul recepţionat este suprascris cu 0 pentru recepţie cu paritate impară şi cu 1 pentru eroare.

*

* *

Pentru a rămîne cu o privire de ansamblu în tabelul 2.4 se prezintă succint dar împreună întregul tablou al registrelor de configurare al microcalculatorului Z8.

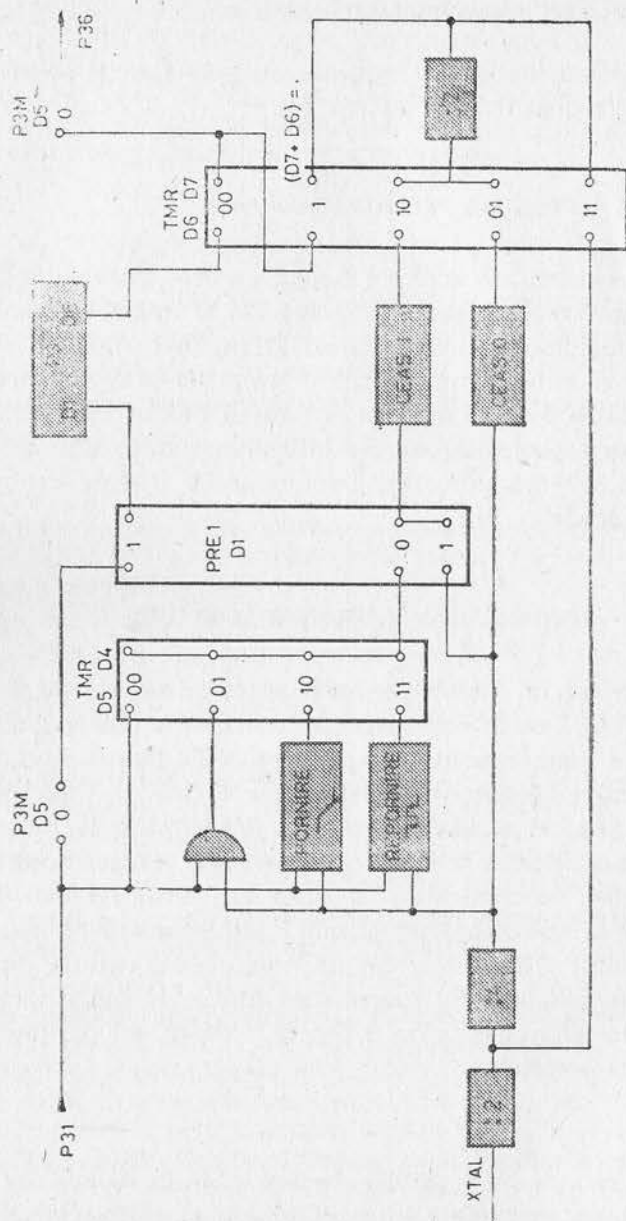


Fig. 2.26. Diagrama simbolică a conexiunilor interne programabile aferente circuitelor de ceas

Tabelul 2.4. ANSAMBLUL REGISTRELOR DE CONFIGURARE ŞI CONTROL

			D7	D6	D5	D4	D3	D2	D1	D0	
SPL	R15	R255	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	
SPH	R14	R254	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	
RP	R13	R253	V7	V6	V5	V4					
FLAGS	R12	R252	CARRY	ZERO	SIGN	OVERFL	D ADJ	M CARRY	UF	UF	
IMR	R11	R251	1 - valid intrerp.		MIRQ5	MIRQ4	MIRQ3	MIRQ2	MIRQ1	MIRQ0	*
IRQ	R10	R250			IRQ5	IRQ4	IRQ3	IRQ2	IRQ1	IRQ0	
IPR	R9	R245									
P01M	R8	R248	P04-07 00 – output 01 – input IX – A8-15		Extended memory timing 1-extended	P10-17 00 – byte out 01 – byte inp 10 AD0-7 11 – SIR		stack set 0 – ext 1 – int	P00 03 00 – output 01 – input IX – A8 – A11		
P3M	R7	R247	PARITY 0 – ØFF 1 – ON	P30 P37 0 inp out 1 SI SO	P31 P36 0 TIN Tout 1 DAV2RDY2	00 01 10 11 P33 inp inp DAV1	P34 out DM RDY1	P32 P35 0 inp out 1 DAV0RDY0		P2M 0 – open d 1 – pull up	
P2M	R6	R246	P27	P26 0 output	P25	P24	P23 1 input	P22	P21	P20	
PREØ	R5	R245	PREØ5	PREØ4	PREØ3	PREØ2	PREØ2	PREØ1		1 - modulo n 0 - num sing	
TØ	R4	R244	T07	T06	T05	T04	T03	T02	T01	T00	**
PRE1	R3	R243	PRE15	PRE14	PRE13	PRE12	PRE11	PRE10	1 - T1 intern 0 - T1 extern	1 - modulo n 0 - num sing	
T1	R2	R242	T17	T16	T15	T14	T13	T12	T11	T10	
TMR	R1	R241	Tout 01 – T0 out 01 – T1 out 11 – int clock out		TIN 00 – ext clock inp 01 – gate inp 10 – trig inp 11 – trig inp retrig		1 enable T1	1 load T1	1 enable T0	1 load T0	
SIO	R0	R240	D7	D6	D5	D4	D3	D2	D1	D0	***
		R127	D7	D6	D5	D4	D3	D2	D1	D0	
		R4									
P3		R3	P37	P36	P35	P34	P33	P32	P31	P30	
P2		R2	P27	P26	P25	P24	P23	P22	P21	P20	
P1		R1	P17	P16	P15	P14	P13	P12	P11	P10	
P0		R0	P07	P06	P05	P04	P03	P02	P01	P00	

* 1 nemascat

** citire - valoare curenta numarator, scriere, incarcare - val initiala

*** citire - receptie data serie, scriere - transmitere data serie

2.8.6. STAREA INIȚIALĂ A REGISTRELOR DE CONFIGURARE ȘI CONTROL

Cu ocazia prezentării fiecărui registru s-au făcut referiri la starea inițială a registrelor. În tabelul de mai jos se prezintă aceste informații în mod condensat.

Tabelul 2.5. VALORILE CU CARE SE INIȚIALIZEAZĂ REGISTRELE DE CONFIGURARE ȘI CONTROL

REGISTRUL	D7	D6	D5	D4	D3	D2	D1	D0	OBSERVAȚII
R240 (SIO)			nedefinit						
R241 (TMR)	0	0	0	0	0	0	0	0	Stop T0, T1
R242 (T1)			nedefinit						
R243 (PRE1)	X	X	X	X	X	X	0	0	— numărare continuă — tact intern
R244 (T0)			nedefinit						
R245 (PRE0)	X	X	X	X	X	X	X	0	— numărare continuă
R246 (P2M)	1	1	1	1	1	1	1	1	— P2 intrare
R247 (P3M)	0	0	0	1	0	0	X	0	— P2 ieșire drena go
R248 (P01M)	0	1	0	0	1	1	0	1	— P3 intrare — P0,P1 — intrare, stiva int.
R249 (IPR)			nedefinit						
R250 (IRQ)	X	X	0	0	0	0	0	0	— cererile de intr.
R251 (IMR)	0	X	X	X	X	X	X	X	— intr. dezact. glob
R252 (Flags)			nedefinit						
R253 (RD)			nedefinit						
R254 (SPH)			nedefinit						
R255 (SPL)			nedefinit						

Din parcurgerea tabelului se observă că toate registrele cu funcții de configurare sînt definite. Registrele care nu sînt implicate în configurare, conținutul lor fiind strict legat de ceea ce dorește să facă utilizatorul se vor înscrie imediat după inițializare prin program de către utilizator.

2.9. ALIMENTAREA ȘI CONTROLUL MICROCALCULATORULUI

2.9.1. OPȚIUNEA „CONSUM REDUS"

Prezența acestei opțiuni se face pe seama sacrificării oscilatorului integrat; în acest caz, tactul sistemului venind pe intrarea XTAL1 de la un oscilator extern. Opțiunea „CONSUM REDUS" permite realimentarea microcalculatorului integrat Z8 fără ca registrele de uz general să-și piardă conținutul. Pentru păstrarea conținutului registrelor generale la căderea tensiunii de alimentare sau în cazurile de oprire intenționată a alimentării, trebuie respectate următoarele condiții:

— intrarea de $\overline{RESET}$ trebuie trasă la masă după salvarea datelor în registrele generale și în timpul dispariției tensiunii de alimentare;

— intrarea VMM/XTAL2 (-pinul 2 la Z8/40 și pinul 63 la Z8/64) se ține la o tensiune minimă de 3V după căderea alimentării;

— intrarea de RESET se ține jos în timpul stabilirii tensiunii de alimentare. După ce tensiunea de alimentare atinge valoarea minimă prescrisă, intrarea de RESET poate lua valoarea de pauză. În timpul funcționării normale, intrarea VMM se ține la același nivel cu tensiunea de alimentare. Jocul tensiunilor Vcc și VMM, necesare prezervării datelor în registrele de uz general, în cazul căderii tensiunii de alimentare și regăsirii informațiilor la realimentarea circuitului Z8, este asigurat de schema din fig. 2.27. Schema este recomandată de firma ZILOG [2].

Căderea tensiunii de alimentare trebuie detectată suficient de devreme, în cazurile în care trebuiesc salvate date, altele decît cele din registrele de uz general, deoarece din setul de registre (144 octeți) numai registrele de uz general (124) sînt alimentate de pe baterie. Detectarea căderii tensiunii urmînd să se facă în exterior. O schemă posibilă de detectare se prezintă în fig. 2.28. Ieșirea comparatorului rapid se pune pe o întrerupere care s-a pus în lanțul de priorități pe poziția cea mai avansată. Condiția primordială pentru implementarea unei proceduri de salvare/refacere în cazul căderii tensiunii de alimentare este ca stiva să fie implementată în

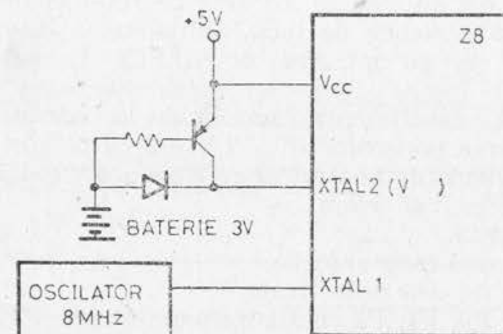


Fig. 2.27. Circuit de alimentare în cazul opțiunii „consum redus"

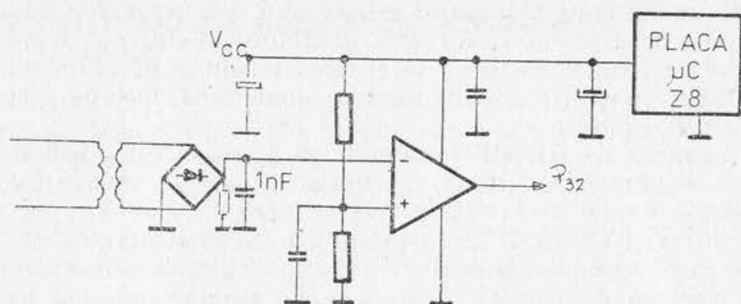


Fig. 2.28. Circuit pentru sesizarea căderii tensiunii

registrele de uz general. Subrutina de tratare a întreruperii trebuie să:

— salveze însăși indicatorul de stivă,

— alte date considerate necesare reluării programului din punctul dinaintea întreruperii,

— salvarea indicatorului de stivă și a celorlalte date să se facă la adrese fixe cunoscute (secvenței de reset).

2.9.2. INIȚIALIZAREA

Inițializarea microcalculatorului Z8 se face folosind intrarea RESET. Inițializarea se face fie la punerea sub tensiune, fie în timpul funcționării. La punerea sub tensiune intrarea RESET trebuie ținută jos cel puțin 50 ms după ce tensiunea de alimentare a atins valoarea minimă admisă (4,75 V), pentru inițializare. Inițializarea în timpul funcționării impune ca intrarea de reset să fie ținută jos cel puțin 4,5 μs (18 perioade de tact). Inițializarea începe după revenirea tensiunii de pe intrarea de RESET la valoarea UIH și constă în:

— începerea execuției programului de la locația 000CH, și

— configurarea porturilor P0, P1 și P2 ca porturi de intrare.

Celelalte registre de control și configurare după inițializare vor fi înscrise astfel încît să avem:

— stiva internă,

— ciclul normal (neprelungit),

— circuitele de ceas dezactivate,

— porturile P0, P1, P2 configurate ca intrări (P3 nu face obiectul configurării),

— cererile de întrerupere șterse și

— întreruperile dezactivate global.

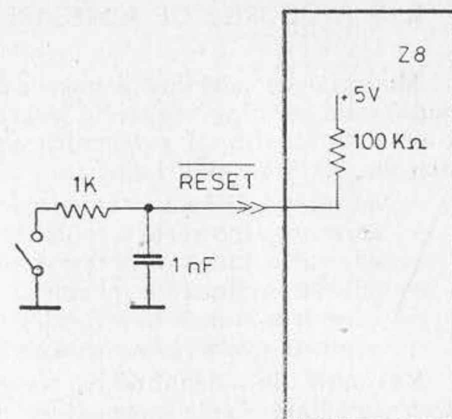


Fig. 2.29. Circuit pentru inițializare automată la punerea sub tensiune

Valorile exacte înscrise în registrele de configurare au fost prezentate la sfîrșitul paragrafului 2.8. Inițializarea la punerea sub tensiune este ușurată de existența unei rezistențe integrate (interne) de 100 KΩ, pe intrarea de RESET. Aceasta permite ca punînd un condensator de 1 μF pe această intrare să se obțină întîrzierea necesară (v. fig. 2.29). În timp ce linia RESET e ținută jos, ieșirea AS generează o frecvență în ritmul tactului intern, ieșirea DS (negat) e ținută jos iar R/W e ținută sus. Deoarece ieșirile AS și DS nu sînt niciodată împreună la 0 în timpul funcționării, această coincidență se poate exploata ca o condiție de inițializare pentru alte componente. Formele de undă corespunzătoare inițializării sînt prezentate în fig. 2.30.

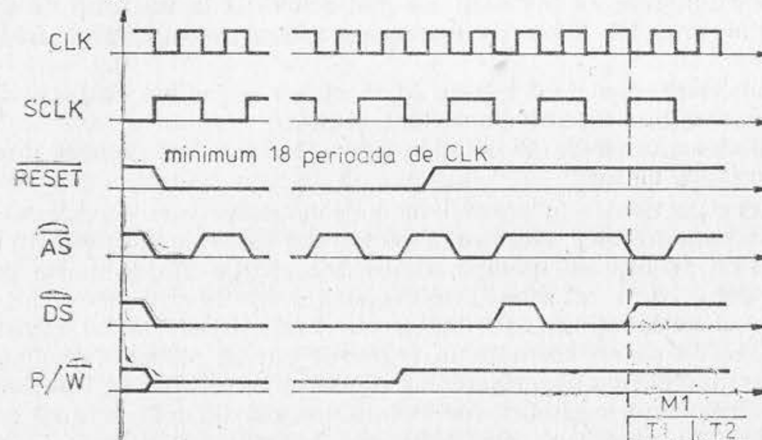


Fig. 2.30. Formele de undă asociate inițializării

2.10. MODURILE DE ADRESARE

Modurile de adresare a microcalculatorului integrat se referă la modul cum se ating registrele interne, locațiile de memorie program și de date în timpul prelucrării secvențiale a instrucțiunilor unui program. Aceste moduri sînt:

— adresarea directă a registrelor
— adresarea indirectă a registrelor
— adresarea indexată a registrelor
— adresarea directă a memoriei
— adresarea indirectă a memoriei
— adresarea relativă a memoriei

Mai mult din obișnuință, se acceptă ca mod de adresare și „adresarea" imediată. Acele instrucțiuni care au operandul în chiar corpul instrucțiunii, sînt referite ca instrucțiuni cu adresare imediată.

2.10.1. ADRESAREA DIRECTĂ A REGISTRELOR

În acest mod de adresare, valoarea operandului sau a operanzilor este dată de conținutul registrului sau registrelor specificate în instrucțiune. Registrele pot fi localizate fie printr-o adresă de opt biți în spațiul setului de registre 0—127 și 240—255, fie printr-o adresă de 4 biți în spațiul 0—15 a unui grup de registre de lucru, grup de registre de lucru specificate prin semioctetul superior al registrului RP. Adresarea unui registru printr-o adresă de 4 biți conduce la o lungime de instrucțiune mai scurtă și la un timp de execuție mai mic. Să luăm de exemplu, adunarea registrelor R04 și R05, prin:

— instrucțiunea de 2 octeți ADD r4, r5 — adună R04 cu R05, dacă RP este 0, în 6 perioade de tact, și prin

— instrucțiunea de 3 octeți ADD R4, R5 face același lucru în 10 perioade de tact.

Un caz particular al acestui mod de adresare este adresarea registrelor pare (duble). Adresarea registrelor pare se face pentru un operand de 16 biți sau pentru a adresa o locație din memoria program. Faptul că ne referim la un registru dublu rezultă din codul de operație al instrucțiunii. Pentru a ne referi corect la un registru dublu trebuie să ne adresăm la registrul par al perechii. Registrul par păstrează partea superioară iar registrul impar partea inferioară a operandului sau a adresei; (de exemplu registrul #21 conține partea inferioară a adresei specificate de registrul dublu #20 — perechea fiind #20, #21.)

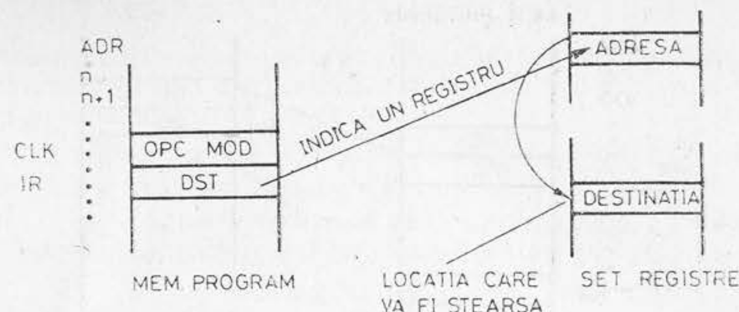


Fig. 2.31. Adresarea indirectă a registrelor

2.10.2. ADRESAREA INDIRECTĂ A REGISTRELOR

În acest mod de adresare, operandul nu este conținutul „registrului"; „registrul" desemnat în corpul instrucțiunii care apelează la adresarea indirectă, va indica un alt registru (o locație de memorie) care conține operandul cu care va opera instrucțiunea în cauză. În definiția de mai sus, prin „registru" trebuie înțeles registru, dar și registru de lucru, și registru pereche și registru pereche de lucru. În continuare vom da un exemplu de adresare indirectă:

— CLR IR — șterge registrul indicat de conținutul registrului în corpul instrucțiunii (v. fig. 2.31)

2.10.3. ADRESAREA INDIRECTĂ A MEMORIEI

Adresarea indirectă a memoriei se face *via* registrelor, adică în corpul instrucțiunii de adresare indirectă a memoriei se specifică un registru dublu, a cărui conținut (16 biți) indică locația de memorie. În continuare vom da un exemplu:

— JP IRR — adresa (operandul) care se va înscrie în contorul PC, ca urmare a execuției instrucțiunii, se găsește în registrul dublu specificat în corpul instrucțiunii RR. Prin urmare, dacă R10=01 și R11=00 (RR10=0100H) atunci instrucțiunea JP IRR10 după execuție va duce microcalculatorul integrat la adresa 0100H de unde va prelua și execuția în continuare a programului (v. fig 2.32).

Un alt caz de adresare indirectă este acela în care ambii operanzi sînt adresați indirect. Fie instrucțiunea de transfer de bloc LDCI, Ir, Irr; prin această instrucțiune încărcăm o zonă de registre specificate de conținutul registrelor Ir (DST)+RP, cu valoarea din lo-

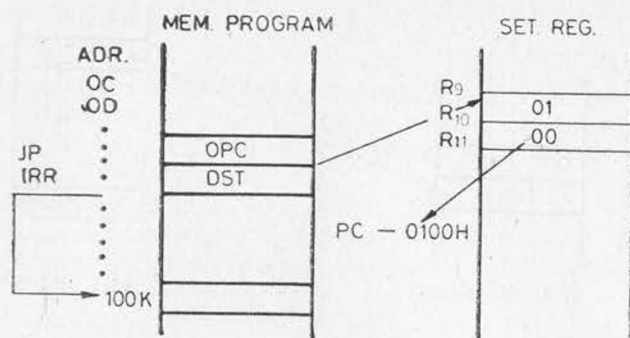


Fig. 2.32. Adresarea indirectă a memoriei program

cațiile de memorie externă adresate de conținutul registrelor duble Irr (SRC)+RP. Fiecare operație de transfer este succedată automat de două operații de incrementare a conținutului registrelor r (DST) și rr (SRC). Acest caz de adresare indirectă este prezentat în figura 2.33.

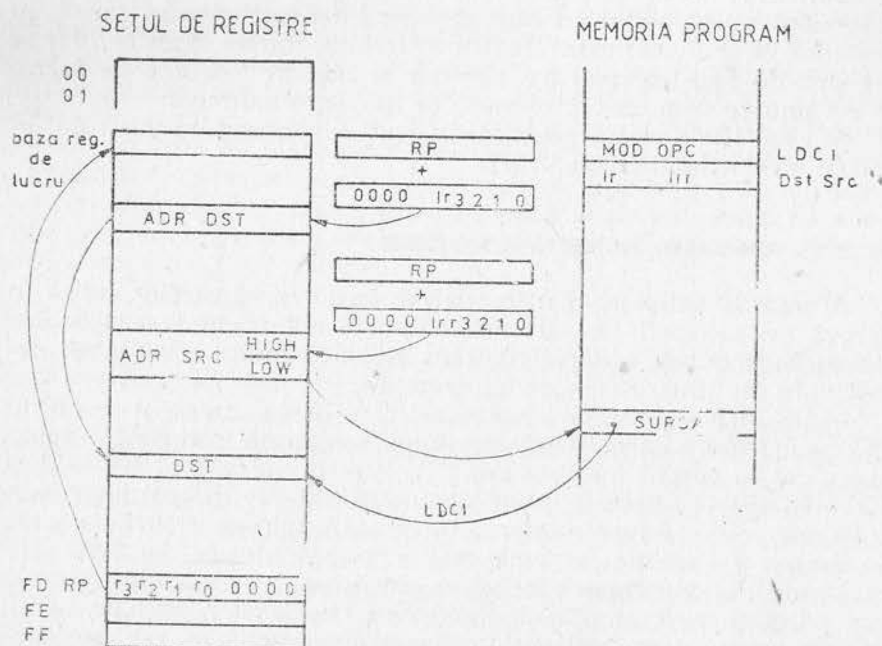


Fig. 2.33. Adresarea indirectă dublă

2.10.4. ADRESAREA INDEXATĂ

Acest mod de adresare este utilizat în numai două instrucțiuni de încărcare pentru a adresa un registru din set. În corpul instrucțiunii se specifică:

1) un registru de lucru care va conține deplasamentul (offsetul),

2) o adresă de bază (indexul) și

3) un registru de lucru care e sursa sau destinația operației. Cele două instrucțiuni și formatul lor sînt prezentate mai jos:

LDr, X + (r')

OPC	MOD
dst	offset
X	

și respectiv

LD X + (r'), r

OPC	MOD
SRC	offset
X	

Luăm în considerare un exemplu concret în condițiile:

— registrul indicator RP indică ultima grupă F,

— se folosește stiva internă, prin urmare registrul R254 (r 14 din grupa de lucru F) se poate folosi ca registru general,

— caracterele recepționate serie se depun în zona R50—R50+m,

— recepția serie se tratează pe întreruperi.

Dacă folosim R254 (r 14 din grupa de lucru F) ca registru ce ține deplasamentul, atunci depunerea caracterului recepționat în registrul R240 (SIO) în zona de registre începînd cu registrul 50 se poate face prin instrucțiunea de adresare indexată LD 50+ (r 14), r 0 introdusă în următoarea secvență de tratare a întreruperii de la recepția serie:

```
SI (IRQ): LD 50 + (r 14), r 0
          INC r 14
          IRET
```

În fig. 2.34 se prezintă modul de adresare indexată pe acest caz.

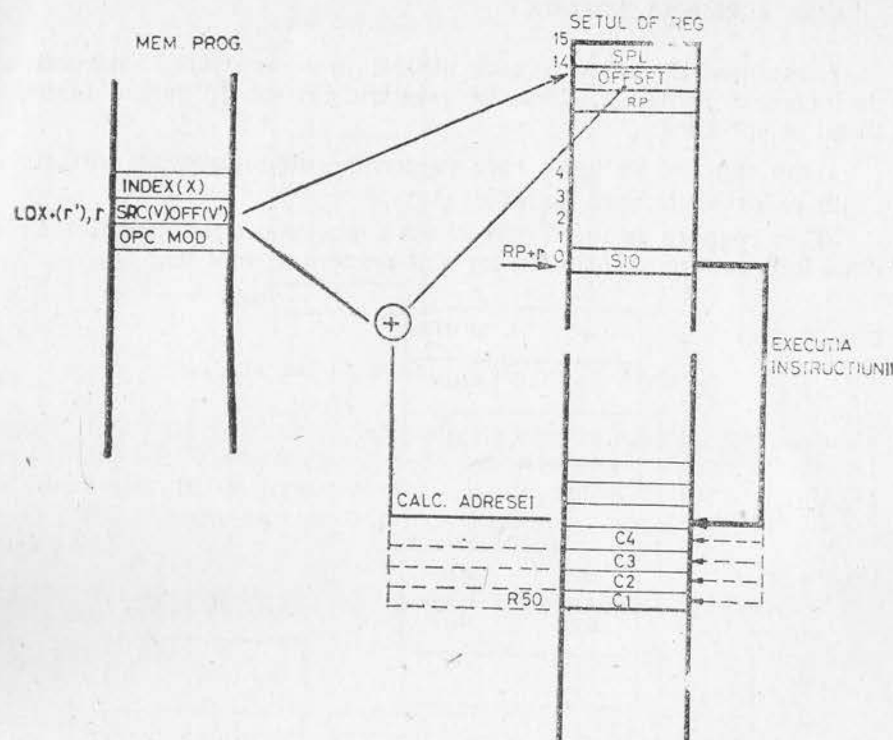


Fig. 2.34. Exemplu de adresare indexată

2.10.5. **ADRESAREA DIRECTĂ**

Adresarea directă presupune că noua adresă se află chiar în corpul instrucțiunii. Acest mod de adresare se folosește în cazul instrucțiunilor de salt și salt condiționat (JP) și al instrucțiunii de chemare subrutină (CALL). Corpul acestei instrucțiuni conține adresa (16 biți) de destinație. În fig. 2.35 este prezentată o secvență program ce cuprinde două exemple de adresare directă și anume:

CALL 0100

JP 2000.

2.10.6. **ADRESAREA RELATIVĂ**

Adresarea relativă este implicată în două tipuri de instrucțiuni:

— instrucțiunile de salt relativ (JR)

— instrucțiunea de decrementare și salt relativ pe condiția non zero. Operandul acestei instrucțiuni conține o adresă relativă față de contorul program curent. Adresele relative negative se dau în complement față de 2. Dacă adresa relativă indică o adresă din față (mai mică), aceasta adăugată contorului program dă adresa următoare a contorului program. Valoarea contorului program luată în discuție e adresa următoarei instrucțiuni după JR, respectiv DJNZ. Adresa relativă este un număr în domeniul — 128 — +127.

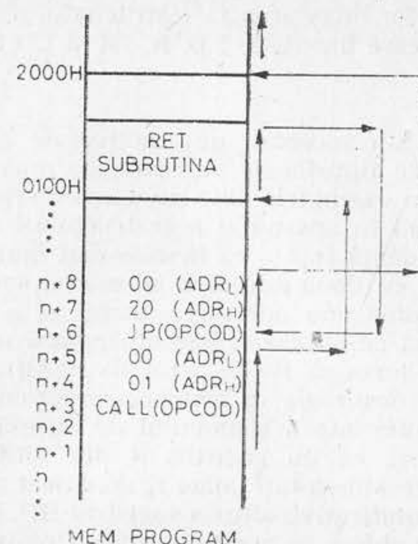
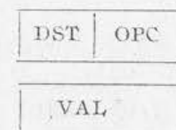


Fig. 2.35. Adresarea directă

2.10.7. **ADRESAREA IMEDIATĂ**

În adresarea imediată operandul se găsește în chiar corpul instrucțiunii. Operandul găsindu-se după codul de operare, în memoria program, adresa sa se află ușor incrementînd contorul program. Acest mod de adresare rezultînd din codul instrucțiunii, ca de altfel toate celelalte așa cum s-a subliniat de fiecare dată, adresarea sau preluarea operandului decurge de la sine. Adresarea imediată se practică îndeosebi la încărcarea registrelor cu valori inițiale. Prin mecanismul său de adresare pe 4 biți, microcalculatorul integrat Z8 dispune de o instrucțiune pe 2 octeți prin care poate încărca orice registru cu o valoare imediată. Această instrucțiune este LD r, IM avînd structura:



Pe lîngă această instrucțiune mai există două instrucțiuni de încărcare imediată: LD, R, IM și LD IR, IM.

* * *

S-a încercat, din motive de claritate, să se prezinte pe rînd toate tipurile de adresare ale microcalculatorul integrat Z8. Totuși prin exemplele date nu s-a urmărit numai adresarea în sine ci execuția în ansamblu a instrucțiunii. Și cu această ocazie s-a pus în evidență faptul că în cele mai multe cazuri avem de-a face de obicei cu două adresări: adresarea sursei și adresarea destinației. Nu întotdeauna adresarea sursei și a destinației se face după același mod de adresare. Sau chiar dacă se face după același mod se poate ca sursa să fie în setul de registre, ca atare adresa ei e pe 8 biți, iar destinația în memoria program avînd adresa pe 16 biți. O particularitate a sistemului de adresare a microcalculatorului Z8 este aceea că un registru R din setul de registre specificat printr-o adresă de 8 biți poate fi specificat și numai prin 4 biți, cei mai puțin semnificativi, și prin registrul RP. Pentru ca microcalculatorul Z8 să asambleze în acest fel adresa registrului, pe poziția celor mai semnificativi 4 biți (din adresa registrului R) se pune invariabil EH. În consecință, apelul la acest mecanism se face specificînd adresa astfel:

1	1	1	0	r3	r2	r1	r0

Acest mecanism e bine să fie cunoscut, cu atît mai mult cu cît sînt și crosasambloare care țin cont de el, pentru a nu crea suspiciuni vizavi de cunoașterea microcalculatorului Z8. În sine, nu reprezintă nimic în plus față de cele prezentate. De exemplu, salvarea în stivă a registrului R21H se poate face prin instrucțiunea PUSH R astfel:

PUSH R	OPC	MOD	70H
	0010	0001	21H

dar și

OPC	MOD	7OH
1110	0001	E1H

cu condiția ca registrul RP să conțină 02H. Această formă nu poate fi asimilată cu o ipotetică instrucțiune PUSH r întrucît codul rămîne tot de octeți. Sau un alt exemplu:

incrementarea unui registru R9:

INC R9	OPC	MOD	20H 09H
	00001001		

care poate fi dată și în forma:

OPC	MOD	20H E9H
1110	1001	

Instrucțiunea de incrementare, cunoaște și o formă care apelează la mecanismul de adresare pe 4 biți și care este foarte compactă. Această formă este:

INC r	DST	OPC	rE

și ocupă numai un octet.

2.11. SETUL DE INSTRUCȚIUNI

2.11.1. STRUCTURA INSTRUCȚIUNILOR

Pentru simplificarea utilizării și înțelegerii instrucțiunilor microcalculatorului Z8 vom recurge la anumite notații, în general folosite, totuși trebuie avut grijă întrucît diverși proiectanți de instrumente software au folosit notații particulare. În ceea ce ne privește am mers pe notațiile firmei Zilog. Cu diferite ocazii, cu unele notații ne-am mai întîlnit, care fie au fost explicate fie s-au subînțeles. Se prezintă, totuși, ansamblul abrevierilor cu semnificațiile lor.

SIMBOL EXPLICAȚIE

r — adresa registrului de lucru (0—15)

R — adresa registrului (0—127, 240—255) sau adresa registrului de lucru

Ir — adresa registrului de lucru care conține adresa de destinație sau sursa
IR — adresa registrului care conține adresa de destinație sau sursa
RR — adresa registrului par sau adresa registrului par de lucru
IRR — adresa registrului par care conține o adresă de memorie
Irr — adresa registrului de lucru par care conține o adresă de memorie
X — adresa registrului de index
DA o adresă de memorie pe 16 biți (HL)
RA — o adresă relativă (—128 — +127)
IM — o valoare pe 8 biți (VAL)
DST — adresa de destinație
SRC — adresa sursei
CC — codul condiției
SP — registrul indicator al stivei, sau locația stivei
PC — numărătorul program sau conținutul acestuia
FLAGS — indicatorii de stare sau registrul indicatorilor de stare
RP — indicatorul de regiștri
IMR — registrul de mascare al întreruperilor
OPC — codificarea operației
MOD — codificarea modului de adresare

Instrucțiunile microcalculatorului Z8 sînt organizate pe 1, 2 sau 3 octeți. În general, forma lor respectă structura: cod de operație generalizat/operand (adresa). Codul de operație, OPC, are o lungime de 8 sau chiar de 4 biți. Primul octet al instrucțiunilor conține obligatoriu codul de operație. În cazul codului de operație de 4 biți, ceilalți 4 biți se referă fie la modul de adresare (MOD) fie la condițiile de salt (CC), fie la adresa unui registru de lucru.

Prin codul de operație generalizat se pot codifica cel mult 256 de operații. Microcalculatorul Z8 dispune de un total de 231 de coduri de operații care provin din 43 de operații de bază care se multiplică fie după diferitele moduri de adresare, fie după condițiile de salt. Partea de operand sau adresă poate fi de maximum 2 octeți funcție de modul de adresare. În continuare vom prezenta structura tuturor instrucțiunilor ordonate pe 1, 2 sau 3 octeți.

INSTRUCȚIUNI DE 1 OCTET

1) | OPC | CCF, SCF, RCF, DI, EI, IRET, RET, NOP

2) | DST | OPC | INC r

INSTRUCȚIUNI DE 2 OCTEȚI

1) | OPC | MODE | CLR, COM, DEC, DECW, INC, INCW, PUSH, RL, RLC, RR, RRC, SRA, SWAP
| DST | (SRC) | sau | 1110 | DST (SRC) |

2) | OPC | JP, CALL (indirect)
| DST | sau | 1110 DST |

3) | OPC | SRP
| VALUE |

4) | OPC | MOD | ADC, ADD, AND, OR, SBC, SUB, TCM, TM, XOR
| DST | SRC |

5) | MOD | OPC | LDC, LDCI, LDE, LDEI, LDr, Ir
| DST | SRC |

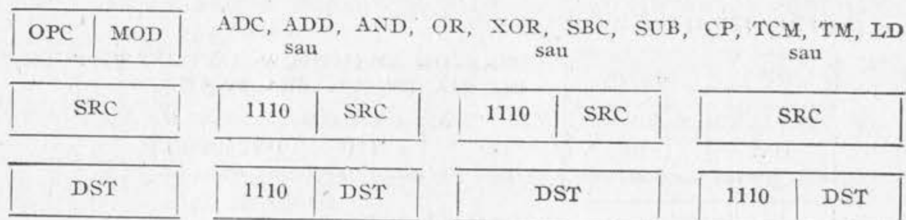
6) | DST(SRC) | OPC | LDr, R, LDr, r
| SRC (DST) |

7) | DST | OPC | LDr, IM
| VALUE |

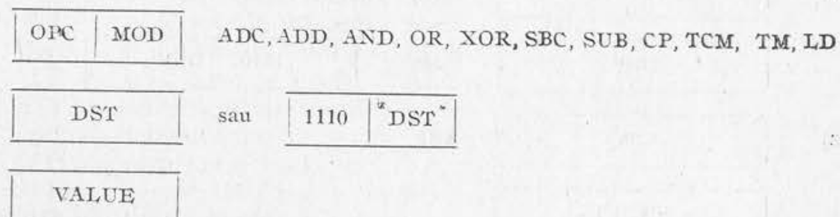
8) | DST(CC) | OPC | DJNZ, JR
| RA |

INSTRUCȚIUNI DE 3 OCTEȚI

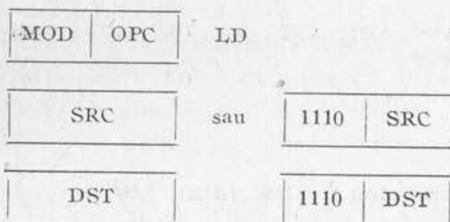
1)



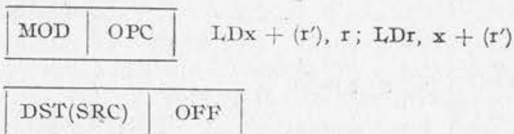
2)



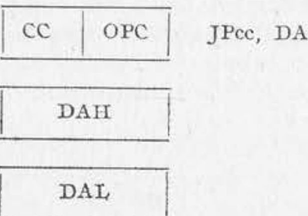
3)



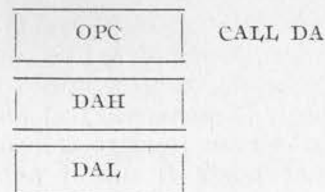
4)



5)



6)



2.11.2. TIPURILE DE INSTRUCȚIUNI

Din punct de vedere funcțional instrucțiunile se împart în instrucțiuni de:

— încărcare sau încărcare repetată
— rotație sau deplasare
— salt (controlul programului)
— test pe bit
— controlul unității centrale
— aritmetice sau logice

Instrucțiunile microcalculatorului integrat Z8 sînt asemănătoare cu cele ale binecunoscutului microprocesor de uz general de 8 biți Z80, însă nu poate fi vorba de compatibilitate în primul rînd din cauza structurii hard diferite. Diferențele constau în modul de organizare și utilizare a registrelor de configurare și control. Fiecare din cele 124 de registre generale poate fi acumulator, adresa, adresa indirectă, registru de index, suport pentru stivă. Prin urmare setul de instrucțiuni Z8 oferă o mai mare suplețe și flexibilitate. Este de așteptat ca viitoarele tipuri de microprocesoare și microcalculatoare integrate să-și asume acest concept de acumulator generalizat.

2.11.3. INDICATORII DE STARE

Instrucțiunile poziționează, funcție de tipul instrucțiunii și al rezultatului, un număr de 6 indicatori de stare necesari controlului programului. Prin instrucțiunile de salt condiționat se iau în considerare acești indicatori fie separat fie în anumite combinații. De asemenea, acești indicatori sînt păstrați și în registrul R252 (FLAGS) și pot fi testați și utilizați și în afara instrucțiunilor de salt. Exact care instrucțiuni ce indicatori afectează, va apare în tabelele de prezentare a instrucțiunilor (tab. 2.7.), deci nu vor mai fi enumerate și aici. Modul cum o instrucțiune afectează sau nu indicatorii de stare este marcat prin următoarele simboluri:

* — modificat conform rezultatului
— — neafectat

0 — șters
1 — înscris
x — necontrolat

C(CARRY) — TRANSPORT. Indicatorul C memorează al nouălea bit generat într-o operație aritmetică într-un registru folosit ca acumulator. De asemenea, acest indicator poate fi folosit pentru testarea unui bit oarecare prin utilizarea instrucțiunilor de rotație și deplasare.

Z(ZERO). Indicatorul de stare Z este afectat de rezultatul 0 al operațiilor aritmetice, logice și de test. Indicatorul Z devine 1 dacă rezultatul operațiilor amintite este 0.

S(SIGN) — SEMN. Indicatorul S este afectat de instrucțiunile aritmetice, logice, de test, de rotație și deplasare. Acest indicator se înscrie, S=1, cînd bitul al 8-lea din registru folosit ca acumulator devine 1; este în fond oglinda acestui bit al rezultatului. În operațiile aritmetice în care numerele negative se scriu în complement față de 2, acest bit indică dacă rezultatul e negativ (S=1) sau pozitiv (S=0).

V(OVERFLOW) — DEPĂȘIRE. Depășirea înseamnă că un transport aritmetic din interiorul octetului a modificat valoarea celui mai semnificativ bit, adică al bitului de semn păstrat, așa cum am arătat, de indicatorul S (sign).

În cazul utilizării complementului față de 2 va rezulta în urma depășirii o eroare de semn. Pentru a corecta astfel de erori se testează indicatorul de depășire în vederea corectării rezultatului. Cînd avem depășire, V=1, acesta indică un rezultat (în complement față de 2) eronat, deoarece a depășit domeniul (—128 — +127) ce poate fi exprimat în această relație pe 8 biți.

D(DECIMAL ADJUST) — AJUSTARE ZECIMALĂ. Acest indicator se folosește în cazul în care numere binare pe 8 biți sînt organizate în două numere ZCB (BCD — 4 biți). Deoarece algoritmii de corecție ai aritmeticii BCD sînt diferiți la adunare și scădere acest indicator ajutător specifică dacă ultima instrucție executată a fost adunare (D=0) sau scădere (D=1). Acest indicator nu este prezent la microprocesoarele I8080, M6800 și Z80.

H(HALF CARRY — TRANSPORT LA JUMĂTATE. Și acest indicator este destinat operațiilor în ZCB. Cu scopul de a compactiza informația, un cuvînt de 8 biți cuprinde două cuvinte ZCB. La efectuarea operațiilor în ZCB poate să apară un transport între ultimul bit al primului cuvînt ZCB și primul bit al celui de al doilea cuvînt ZCB din octet. Acest transport trebuie corectat, de aceea indicatorul H se înscrie la detectarea unui transport.

F1 și F2(USER FLAGS) — sînt biții D0 și D1 din registrul

R252 (FLAGS) ei nu sînt afectați de instrucțiuni și nu reflectă nici o stare. Ei pot fi folosiți, adică înscriși și sterși funcție de necesitatea și utilitatea programatorului.

2.11.4. CONDIȚIILE

Instrucțiunile de salt direct și relativ sînt condiționate de valoarea indicatoarelor de stare. Valoarea acestor indicatoare este codificată pe 4 biți, iar aceste condiții apar în codul de operație generalizat al celor două instrucțiuni. Sînt în total 16 astfel de condiții; din considerente de simplitate și simetrie unele condiții sînt dublate astfel că apar toate cele 20 de forme, recunoscute ca atare și de crosasambloare. De exemplu avem condiția NZ, dar și EQ pentru aceeași valoare a indicatorului Z (Z=1). Codurile de condiție identice sînt:

Z și EQ pentru Z=1 0110
NZ și NE pentru Z=0 1110
C și ULT pentru C=1 0111
NC și UGE pentru C=0 1111

În tabelul nr. 2.6. se prezintă cele 16 coduri de condiție în cele 20 de forme în care ele pot să apară:

Tabelul nr. 2.6. CODURILE DE CONDIȚIE

CONDIȚIA DE SALT	CODUL		POZIȚIA INDICATORILOR	SEMNIFICAȚIE
	BINAR	HEXA		
	0000	0		totdeauna fals
	1000	8		totdeauna adevărat
Z	1010	6	Z=1	zero
NZ	1110	E	Z=0	diferit de zero
C	0111	7	C=1	transport
NC	1111	F	C=0	fără transport
MI	0101	5	S=1	minus
PL	1101	D	S=0	plus
OV	0100	4	V=1	depășire
NOV	1100	C	V=0	fără depășire
EQ	0110	6	Z=1	egal
NE	1110	E	Z=0	diferit
LT	0001	1	S XOR V=1	mai mic decît
GE	1001	9	S XOR V=0	mai mare sau egal
LE	0010	2	Z OR (S XOR V)=1	mai mic sau egal
GT	1010	A	Z OR (S XOR V)=0	mai mare decît
ULT	0111	7	C=1	mai mic decît '
UGE	1111	F	C=0	mai mare sau egal '
ULE	0011	3	(C OR Z)=1	mai mic sau egal '
UGT	1011	B	C=0 și Z=0	mai mare decît '.

Condiţiile cu prefixe U (unsigned) se referă la numerele binare pe 8 biţi, pe cînd celelalte se referă la numerele reprezentate în complementul faţă de 2.

2.11.5. SETUL DE INSTRUCŢIUNI. CODURILE

Pentru a înţelege tabelul şi mai departe instrucţiunile, trebuie lămurit faptul că s-a făcut un compromis în prezentarea instrucţiunilor. Acest compromis constă în faptul că deşi modul de adresare apare în acelaşi octet cu codul de operaţie s-a convenit ca în prezentare modul de adresare să însoţească şi destinaţia şi sursa. Fapt ce asigură unitate modului de prezentare a instrucţiunilor. Astfel codurile:

20 07 (INC R) — reprezintă incrementarea reg. 07

21 07 (INC IR) — reprezintă incrementarea reg. indicat de conţinutul registrului 07.

Deci atît R cît şi IR este o adresă; faptul că una trebuie privită ca adresa indirectă rezultă nu din codificarea destinaţiei, ci din codul generalizat al instrucţiei. În acest caz particular diferenţa de adresă rezultă din diferenţa codului de operaţie generalizat 20H — 21H. De asemenea, prin registrul (R) sau registrul par indirect (IRR) — adică în general prin registru — trebuie înţeles pe lîngă registru specificat pe 8 biţi şi registru de lucru specificat pe 4 biţi combinat cu adresa specificată de registrul indicator RP (ceilalţi 4 biţi). Acest lucru a fost pus în evidenţă clar şi cu ocazia prezentării structurii instrucţiunilor şi rezultă şi din prezentarea ce urmează a instrucţiunilor. Atunci cînd se specifică că e vorba de registru de lucru, atunci referirea este strict la acestea.

S-au grupat instrucţiunile în instrucţiuni:

— de încărcare

— aritmetico — logice

— de rotaţie şi deplasare

— de test de bit

— de control a programului şi a unităţii centrale.

În continuare, se vor prezenta toate instrucţiunile microcalculatorului integrat Z8 pe grupe, şi în interiorul grupei în ordine alfabetică. Forma de prezentare a instrucţiunilor este una tabelară din care să rezulte pe lîngă cod şi mnemonica şi structura lor, durata în cicli, indicatoarele de stare afectate. De asemenea, s-a încercat într-o manieră sugestivă, semigrafică, să se prezinte acţiunea propriu-zisă a instrucţiunilor.

Tabelul 2.7. a)

INSTRUCŢIUNI Z8 DE ÎNCĂRCARE

Cod	Mnemonica	Structura instrucţiunii / B1 / B2 / B3 /	C.EX.	Indic. CZSVDH	Observaţii
B0	CLR R	/OPC MOD /DST /	6	— — — —	R <— — 0
B1	CLR IR	/OPC MOD /DST /	6	— — — —	IR <— — 0
C7	LD r,((r')+B)	/MOD OPC /DST OFF /VAL /	10	— — — —	r <— — ((r') + BAZA)
D7	LD ((r')+B),r	/MOD OPC /SRC OFF /VAL /	10	— — — —	((r') + BAZA) <— — r
E3	LD r,Ir	/MOD OPC /DST SRC /	6	— — — —	r <— — Ir
E4	LD R,R'	/OPC MOD /SRC /DST /	10	— — — —	R <— — R'
E5	LD R,IR	/OPC MOD /SRC /DST /	10	— — — —	R <— — IR
E6	LD R,IM	/OPC MOD /DST /VAL /	10	— — — —	R <— — VAL
E7	LD IR,IM	/OPC MOD /DST /VAL /	10	— — — —	IR <— — VAL
F3	LD Ir,r	/MOD OPC /DST SRC /	6	— — — —	Ir <— — r
F5	LD IR,R	/OPC MOD /SRC /DST /	10	— — — —	IR <— — R
r8	LD r,R	/DST OPC /SRC /	6	— — — —	r <— — R
r9	LD R,r	/SRC OPC /DST /	6	— — — —	R <— — r
rC	LD r,IM	/DST OPC /VAL /	6	— — — —	r <— — VAL
C2	LDC r,Irr	/MOD OPC /DST SRC /	12	— — — —	r <— — Irr
D2	LDC Irr,r	/MOD OPC /DST SRC /	12	— — — —	Irr <— — r
C3	LDCI Ir,Irr	/MOD OPC /DST SRC /	18	— — — —	INC. din MEM. in REG. IND. Ir progresiv
D3	LDCI Irr,Ir	/MOD OPC /DST SRC /	18	— — — —	INC. in MEM. din REG. IND., Ir progresiv
82	LDE r,Irr	/MOD OPC /DST SRC /	12	— — — —	INC. DIN MEM. DE DATE IN r
92	LDE Irr,r	/MOD OPC /DST SRC /	12	— — — —	INC. IN MEM. DE DATE DIN r
83	LDEI Ir,Irr	/MOD OPC /DST SRC /	18	— — — —	INC DIN MEM. DE DATE, progresiv
93	LDEI Irr,Ir	/MOD OPC /DST SRC /	18	— — — —	INC IN M.DATE, progresiv

INSTRUCŢIUNI Z8 ARITMETICO-LOGICE

Cod	Mnemonica	Structura instrucţiunii / B1 / B2 / B3 /	C.ex.	Indic. CZSVDH	Observaţii
12	ADC r,r'	/OPC MOD /DST SRC /	6	****0*	r <–– r + r' + C
13	ADC r,Ir	/OPC MOD /DST SRC /	6	****0*	r <–– r + Ir + C
14	ADC R,R'	/OPC MOD /SRC /DST /	10	****0*	R <–– R + R' + C
15	ADC R,IR	/OPC MOD /SRC /DST /	10	****0*	R <–– R + IR + C
16	ADC R,IM	/OPC MOD /DST /VAL /	10	****0*	R <–– R + VAL + C
17	ADC IR,IM	/OPC MOD /DST /VAL /	10	****0*	IR <–– IR + VAL + C
02	ADD r,r'	/OPC MOD /DST SRC /	6	****0*	r <–– r + r'
03	ADD r,Ir	/OPC MOD /DST SRC /	6	****0*	r <–– r + Ir
04	ADD R,R'	/OPC MOD /SRC /DST /	10	****0*	R <–– R + R'
05	ADD R,IR	/OPC MOD /SRC /DST /	10	****0*	R <–– R + IR
06	ADD R,IM	/OPC MOD /DST /VAL /	10	****0*	R <–– R + VAL
07	ADD IR,IM	/OPC MOD /DST /VAL /	10	****0*	IR <–– IR + VAL
52	AND r,r'	/OPC MOD /DST SRC /	6	–**0––	r <–– r AND r'
53	AND r,Ir	/OPC MOD /DST SRC /	6	–**0––	r <–– r AND Ir
54	AND R,R'	/OPC MOD /SRC /DST /	10	–**0––	R <–– R AND R'
55	AND R,IR	/OPC MOD /SRC /DST /	10	–**0––	R <–– R AND IR
56	AND R,IM	/OPC MOD /DST /VAL /	10	–**0––	R <–– R AND VAL
57	AND IR,IM	/OPC MOD /DST /VAL /	10	–**0––	IR <–– IR AND VAL
60	COM R	/OPC MOD /DST /	6	–**0––	R <–– NOT R
61	COM IR	/OPC MOD /DST /	6	–**0––	IR <–– NOR IR
A2	CP r,r'	/OPC MOD /DST SRC /	6	****––	FLAGS <–– r – r'
A3	CP r,Ir	/OPC MOD /DST SRC /	6	****––	FLAGS <–– r – Ir
A4	CP R,R'	/OPC MOD /SRC /DST /	6	****––	FLAGS <–– R – R'
A5	CP R,IR	/OPC MOD /SRC /DST /	10	****––	FLAGS <–– R – IR
A6	CP R,IM	/OPC MOD /DST /VAL /	10	****––	FLAGS <–– R – VAL
A7	CP IR,IM	/OPC MOD /DST /VAL /	10	****––	FLAGS <–– IR – VAL
40	DA R	/OPC MOD /DST /	8	***X––	R <–– CONV.ZEC. (R)
41	DA IR	/OPC MOD /DST /	8	***X––	IR <–– CONV.ZEC. (IR)
00	DEC R	/OPC MOD /DST /	6	–***––	R <–– R – 1
01	DEC IR	/OPC MOD /DST /	6	–***––	IR <–– IR – 1
80	DECW RR	/OPC MOD /DST /	10	–***––	RR <–– RR – 1
81	DECW IRR	/OPC MOD /DST /	10	–***––	IRR <–– IRR – 1
20	INC R	/OPC MOD /DST /	6	–***––	R <–– R + 1
21	INC IR	/OPC MOD /DST /	6	–***––	IR <–– IR + 1
rE	INC r	/DST OPC /	6	–***––	r <–– r + 1
A0	INCW RR	/OPC MOD /DST /	10	–***––	RR <–– RR + 1
A1	INCW IRR	/OPC MOD /DST /	10	–***––	IRR <–– IRR + 1
42	OR r,r'	/OPC MOD /DST SRC /	6	–**0––	r <–– r OR r'
43	OR r,Ir	/OPC MOD /DST SRC /	6	–**0––	r <–– r OR Ir
44	OR R,R'	/OPC MOD /SRC /DST /	10	–**0––	R <–– R OR R'
45	OR R,IR	/OPC MOD /SRC /DST /	10	–**0––	R <–– R OR IR
46	OR R,IM	/OPC MOD /DST /VAL /	10	–**0––	R <–– R OR VAL
47	OR IR,IM	/OPC MOD /DST /VAL /	10	–**0––	IR <–– IR OR VAL
90	RL R	/OPC MOD /DST /	6	****––	
91	RL IR	/OPC MOD /DST /	6	****––	
10	RLC R	/OPC MOD /DST /	6	****––	
11	RLC IR	/OPC MOD /DST /	6	****––	
E0	RR R	/OPC MOD /DST /	6	****––	
E1	RR IR	/OPC MOD /DST /	6	****––	
C0	RRC R	/OPC MOD /DST /	6	****––	
C1	RRC IR	/OPC MOD /DST /	6	****––	
32	SBC r,r'	/OPC MOD /DST SCR /	6	****1*	r <–– r – r' – C
33	SBC r,Ir	/OPC MOD /DST SRC /	6	****1*	r <–– r – Ir – C
34	SBC R,R'	/OPC MOD /SRC /DST /	10	****1*	R <–– R – R' – C
35	SBC R,IR	/OPC MOD /SRC /DST /	10	****1*	R <–– R – IR – C
36	SBC R,IM	/OPC MOD /DST /VAL /	10	****1*	R <–– R – VAL – C
37	SBC IR,IM	/OPC MOD /DST /VAL /	10	****1*	IR <–– IR – VAL – C
D0	SRA R	/OPC MOD /DST /	6	***0––	DEPL. ARITM. DR.
D1	SRA IR	/OPC MOD /DST /	6	***0––	DEPL. ARITM. DR.
22	SUB r,r'	/OPC MOD /DST SRC /	6	****1*	r <–– r – r'
23	SUB r,Ir	/OPC MOD /DST SRC /	6	****1*	r <–– r – Ir
24	SUB R,R'	/OPC MOD /SRC /DST /	10	****1*	R <–– R – R'
25	SUB R,IR	/OPC MOD /SRC /DST /	10	****1*	R <–– R – R
26	SUB R,IM	/OPC MOD /DST /VAL /	10	****1*	R <–– R – IR
27	SUB IR,IM	/OPC MOD /DST /VAL /	10	****1*	IR <–– IR – VAL
F0	SWAP R	/OPC MOD /DST /	8	X**X––	SE SCHIMBA SEMIOCTETII INTRE EI
F1	SWAP IR	/OPC MOD /DST /	8	X**X––	SE SCHIMBA SEMIOCTETII INTRE EI
B2	XOR r,r'	/OPC MOD /DST SRC /	6	–**0––	r <–– r XOR r'
B3	XOR r,Ir	/OPC MOD /DST SRC /	6	–**0––	r <–– r XOR Ir
B4	XOR R,R'	/OPC MOD /SRC /DST /	10	–**0––	R <–– R XOR R'
B5	XOR R,IR	/OPC MOD /SRC /DST /	10	–**0––	R <–– R XOR IR
B6	XOR R,IM	/OPC MOD /DST /VAL /	10	–**0––	R <–– R XOR VAL
B7	XOR IR,IM	/OPC MOD /DST /VAL /	10	–**0––	IR <–– IR XOR VAL

2.7. c)

INSTRUCȚIUNI Z8 DE ROTAȚIE ȘI DEPLASARE

Cod	Mnemonica	Structura instrucțiunii	C.ex. indic. C Z S V D H	Observații
90	RL R	/OPC MOD /DST /	6* * * * – –	
91	RL IR	/OPC MOD /DST /	6* * * * – –	
10	RLC R	/OPC MOD /DST /	6* * * * – –	
11	RLC IR	/OPC MOD /DST /	6* * * * – –	
E0	RR R	/OPC MOD /DST /	6* * * * – –	
E1	RR IR	/OPC MOD /DST /	6* * * * – –	
C0	RRC R	/OPC MOD /DST /	6* * * * – –	
C1	RRC IR	/OPC MOD /DST /	6* * * * – –	
D0	SRA R	/OPC MOD /DST /	6* * * 0– –	DEPL. ARITM. DR.
D1	SRA IR	/OPC MOD /DST /	6* * * 0– –	DEPL. ARITM. DR.
F0	SWAP R	/OPC MOD /DST /	8X* * X– –	SE SCHIMBA SEMIOCTETII INTRE EI
F1	SWAP IR	/OPC MOD /DST /	8X* * X– –	SE SCHIMBA SEMIOCTETII INTRE EI

2.7. d)

INSTRUCȚIUNI Z8 DE TEST

Cod	Mnemonica	Structura instrucțiunii / B1 / B2 / B3 /	C.ex.	Indic. CZSVDH	Observații
62	TCM r,r'	/OPC MOD /DST SRC /	6	– * * 0– –	FLAGS <– – NOT r AND r'
63	TCM r,Ir	/OPC MOD /DST SRC /	6	– * * 0– –	FLAGS <– – NOT r AND Ir
64	TCM R,R'	/OPC MOD /SRC /DST /	10	– * * 0– –	FLAGS <– – NOT R AND R'
65	TCM R,IR	/OPC MOD /SRC /DST /	10	– * * 0– –	FLAGS <– – NOT R AND IR
66	TCM R,IM	/OPC MOD /DST /VAL /	10	– * * 0– –	FLAGS <– – NOT R AND VAL
67	TCM IR,IM	/OPC MOD /DST /VAL /	10	– * * 0– –	FLAGS <– – NOT IR AND VAL
72	TM r,r'	/OPC MOD /DST SRC /	6	– * * 0– –	FLAGS <– – r AND r'
73	TM r,Ir	/OPC MOD /DST SRC /	6	– * * 0– –	FLAGS <– – r AND Ir
74	TM R,R'	/OPC MOD /SRC /DST /	10	– * * 0– –	FLAGS <– – R AND R'
75	TM R,IR	/OPC MOD /SRC /DST /	10	– * * 0– –	FLAGS <– – R AND IR
76	TM R,IM	/OPC MOD /DST /VAL /	10	– * * 0– –	FLAGS <– – R AND VAL
77	TM IR,IM	/OPC MOD /DST /VAL /	10	– * * 0– –	FLAGS <– – IR AND VAL

2.71 e)

INSTRUCȚIUNI Z8 DE CONTROL

Cod	Mnemonica	Structura instrucțiunii / B1 / B2 / B3 /	C.ex.	Indic. CZSVDH	Observații
D4	CALL IRR	/OPC /DST / /	20	------	SP <—— SP−2, @SP <—— PC, PC <—— IRR
D6	CALL DA	/OPC /ADR H /ADR L /	20	------	SP <—— SP−2, @SP <—— PC, PC <—— DA
EF	CCF	/OPC /	6	*-----	C <—— NOT C
8F	DI	/OPC /	6	------	IMR(7) <—— 0
rA	DJNZ r,RA	/DST OPC /ADR REL/	12	------	r <—— r − 1, DACA r = 0 PC <—— PC + RA
9F	EI	/OPC /	6	------	IMR(7) <—— 1
BF	IRET	/OPC /	16	******	FL <—— @SP, PC <—— @SP, IMR(7) <—— 1,SP LA ZI
30	JP IRR	/OPC /DST /	8	------	PC <—— DST
cD	JP CC,DA	/CON OPC /ADRH /ADRL /	12	------	DACA CC ADEV. PC <—— DA
cB	JR CC,RA	/CON OPC /ADR.REL./	10	------	DACA CC ADEV. PC <—— PC + RA
FF	NOP	/OPC /	6	------	
50	POP R	/OPC MOD /DST /	10	------	R <—— @SP SP, <—— SP + 1
51	POP IR	/OPC MOD /DST /	10	------	IR <—— @SP SP, <—— SP + 1
70	PUSH R	/OPC NOD /SRC /	10/12	------	@SP <—— R, SP <—— SP − 1
71	PUSH IR	/OPC MOD /SRC /	12/14	------	@SP <—— IR, SP <—— SP − 1
CF	RCF	/OPC /	6	0-----	C <—— 0
AF	RET	/OPC /	14	------	PC <—— @PC, SP <—— SP + 2
DF	SCF	/OPC /	6	1-----	C <—— 1
31	SRP IM	/OPC /VAL /	6	------	RP <—— VAL

2.12. ASPECTELE TEHNICE DISCUTABILE ALE MICROCALCULATORULUI Z8

Între aspectele tehnice ale microcalculatorului integrat Z8 care merită atenție din partea oricărui proiectant se numără și:

1) absența regimului de funcționare pas cu pas,
2) sistemul de tratare a întreruperilor,
3) lipsa variantei de dezvoltare Z8/40 cu soclu pe circuit la firmele care reprezintă sursa a doua (Mikroelektronik Erfurt),
4) consumul relativ ridicat și
5) unidirecționalitatea magistralei interne dintre procesor și memoria fixă, în cazul variantei Z8/64.

Absența regimului de funcționare pas cu pas se explică prin faptul să circuitul conține elemente dinamice ce trebuiesc refre-șate; circuitul nu este complet static și mai mult ca sigur este vorba de setul de registre. Lipsa regimului pas cu pas, atît de util în momentul punerii la punct a unui proiect, conduce la un sistem de dezvoltare mai mare, mai elaborat. Circuitul nu poate fi folosit de oricine: folosirea lui impune un anumit nivel de dotare și în plus niște instrumente specifice ca crosasambloare, simulatoare de EPROM, monitoare, simulatoare software și inscriptoare de EPROM-uri. Aceste instrumente specifice reduc din dezavantajul discutat. Dezavantajul s-ar fi pulverizat dacă magistrala internă, procesor-memorie program, ar fi fost bidirecțională în cazul variantei de dezvoltare Z8/64; dar ea a fost proiectată strict pentru cazul concret de citire a memoriei program. Dacă magistrala ar fi fost bidirecțională, în proiectele de dezvoltare în care memoria program ar fi fost implementată în exterior cu memorie RAM (cazul simulatoarelor de EPROM), s-ar fi putut dezvolta monitoare cu posibilități de întrerupere (breakpoint). O soluție alternativă ar fi punerea la punct a proiectelor cu ajutorul simulatoarelor software, care deocamdată la noi încă nu există, nici aduse și nici realizate aici. Se vor aduce sau se vor scrie, dar întrucît între aplicațiile cele mai atractive la care va fi folosit Z8-ul vor fi și aplicațiile în timp real — experimentarea cu ajutorul simulatoarelor se va rezuma la aspectul funcțional, validarea sub aspect dinamic urmînd să se facă tot pe viu. Soluția adoptată de autor la punerea în funcțiune a unui modul de control a unei axe de mișcare dintr-o unitate de comandă robot a fost aceea a adăugării în anumite puncte de interes (în punctele de test, ramificație și salt) a unor subprograme de afișare a stării registrelor după care se revenea sau nu în monitor. Din monitor se putea explora mai în detaliu starea în care se găsește în acel moment ansamblul. Aceste subprograme erau înlăturate în

momentul în care se depășea cîte o subfază de testare și se adăugau pe noile ramuri ce urmau să se testeze, pînă la terminarea punerii în funcțiune și la completa lor eliminare.

Consumul ridicat, aproximativ 1W, este un dezavantaj relativ. Relativ, întrucît se poate vorbi de un dezavantaj numai în contextul în care au apărut alte microcalculatoare integrate echivalente ca performanțe, realizate în tehnologie CMOS rapidă. Totuși pentru că au apărut deja microcontrolere CMOS care consumă doar 10 mW și ținînd cont de evoluția deosebit de rapidă din domeniul tehnologiilor microelectronice, este foarte susceptibilă și apariția microcalculatoarelor integrate CMOS. Ca atare, această caracteristică trebuie avută permanent în vedere.

Faptul că varianta Z8/40 cu soclu pe spatele circuitului nu există asimilată în est, de unde ne aprovizionăm, are repercusiuni negative doar în cazul proiectelor destinate bunurilor de consum sau producției de serie, pentru că ar fi de dorit, în aceste cazuri, ca în produsul final să se găsească circuitul Z8 cu programul „ars" în el. Trebuie spus, însă, că proiectele care se întrevăd, deocamdată, au mai mult caracter de unicat, și de serie mică. Deci folosirea variantei Z8/64 cu EPROM alături, în dezvoltare ca și în produsul final nu reprezintă un dezavantaj.

Sistemul de întreruperi al microcalculatorului integrat Z8 este mai puțin performant, după cum s-a putut vedea, decît cel cunoscut în cazul microprocesorului Z80.

* * *

La noi circuitul Z8 a fost studiat și folosit și la ITC București (col. Traian Ciobanu) și Cluj-Napoca, la IEMI București (Viorel Bălan), la Inst. Politehnic Cluj-Napoca (Dan Roman), la ICSITE (ICE), la Electromagnetica București — unde s-au încercat și pus la punct anumite instrumente de dezvoltare pentru acest microcalculator.

3. MODULE HARD ȘI SOFT DE DEZVOLTARE

În continuare se descriu cîteva „instrumente" necesare proiectării cu microcalculatorul integrat Z8. Se prezintă un modul de dezvoltare, un simulator de EPROM și un monitor. La aceste trei „instrumente" strict necesare dezvoltării de proiecte pe baza microcalculatorului integrat Z8 ar mai fi necesar un crossasamblor. Cum un astfel de program se poate scrie sau chiar genera automat pe de o parte, iar pe de altă parte, prezentarea unui astfel de produs software excede obiectivul cărții, ne mulțumim să semnalăm că un astfel de produs software a fost elaborat încă din 1988 la IEMI București.

3.1. NUCLEUL DE DEZVOLTARE CU Z8

Acest nucleu de dezvoltare a fost prima unealtă realizată de autor pentru înjghebarea minimului necesar asimilării microcalculatorului integrat Z8, în condiții de pionierat, ținînd cont și de sărăcia informației disponibile: un manual de firmă preliminar [2], cu unele greșeli, succint dar incomplet și o documentație [1] prelucrată după un material de la firma ZILOG, mai condensat decît primul material dar care înlătură 2—3 greșeli ale lui. În figura 3.1. se prezintă schema bloc a nucleului de dezvoltare construit în jurul versiunii de dezvoltare (Z8/64). Microcalculatorul integrat are la poarta P0, care poate fi programată pe semiocteți, două circuite 8216 al căror sens de vehiculare a informației poate fi ales funcție de puntea care se pune între pinul 15(DIEN) și masă sau +5V. Portul P1, care se programează în totalitate (pe 8 biți) ca intrare sau ieșire, are ca tampon circuitul bidirecțional 8286. Sensul de vehiculare se alege prin puntea între pinul 11 (T) și masa sau + 5V. Întrucît portul P2 poate fi programat linie cu linie ca intrare sau ieșire, acesta nu are nici un circuit tampon pentru a nu reduce fle-

Fig. 3.1. Nucleu de dezvoltare cu Z8

xibilitatea de utilizare a portului P2. În cazul portului P3 avînd întotdeauna liniile P30—33 intrări și liniile P34—37 ieșiri, folosim ca tampon circuitul 74LS244 după cum se arată în schema electrică de detaliu (fig. 3.2.). Pe magistralele A0—A11 și D0—D7 de conectare a memoriei program sînt rezervate socluri pentru două circuite EPROM tip 2716 sau un circuit de tip 2732. Astfel că nucleul poate fi folosit atît cu microcalculatoare Z8/64 cu memorie de 2ko cît și cu versiunea de 4ko. În fig. 3.2. se prezintă schema electrică de detaliu.

Schema de amplasare și cablajul imprimat al nucleului de dezvoltare sînt redate în figurile 3.3. și 3.4. Placa nucleului de dezvoltare se interfațează cu aplicația ce urmează a se dezvolta printr-un conector de tip EUROCARD 3*32. Este de observat că în cazul unor

Fig. 3.2. Schema electrică de detaliu

aplicații concrete, acest nucleu poate rămîne la nivelul microcalculatorului integrat Z8/40 împreună cu cuarțul și circuitul de inițializare.

3.2. SIMULATORUL DE EPROM

Scrierea unui program de control în timp real, cum sînt majoritatea aplicațiilor microcalculatorului integrat, fără greșeală, este un caz care nu se întîlnește practic și nu merită luat în considerare. În consecință, înscrierea programelor în memorii EPROM care se șterg și se înscriu repetat va încetini ritmul de punere la punct al programelor și deci a întregului proiect. O greșeală la începutul programului, de exemplu în partea de program ce configurează mi-

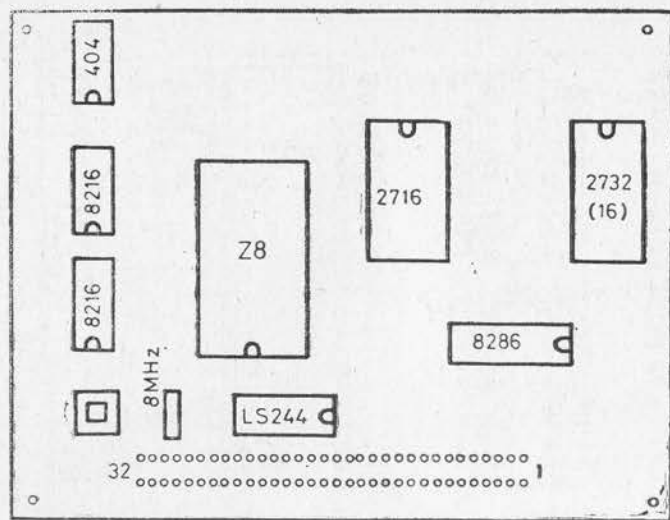
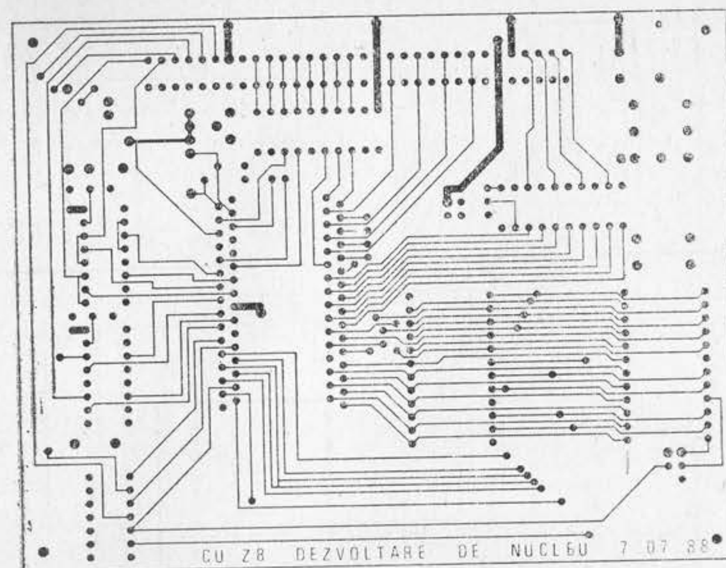


Fig. 3.3. Schiţa de amplasare

crocalculatorul, face imposibilă continuarea depanării programului. Pentru a putea continua EPROM-ul trebuie şters şi reînscris cu corectura necesară. Şi întrucît acest lucru se repetă de zeci de ori, devine clar că o astfel de procedură este mult mai înceată, comparativ cu memoria RAM văzută de microcalculatorul integrat ca memoria sa program, pe de o parte. Pe de altă parte, aceeaşi memorie RAM trebuie să aparţină şi unui microcalculator de dezvoltare. Microcalculator care să dispună de resurse hard şi soft necesare asamblării programelor Z8, modificării şi vizualizării acestora. Această memorie RAM cu acces dublu o numim simulator de EPROM. Schema bloc a simulatorului este prezentată în figura 3.5. Schema electrică a simulatorului este prezentată în figura 3.6.

Implementarea prezentată în figurile 3.5 şi 3.6 are 4ko de memorie RAM realizată cu circuitele 2114 cu scopul de a satisface atît aplicaţiile cu UB8820 cît şi cu UB8840. Un multiplexor 2 la 1 alege liniile de adrese, date şi control între sistemul de dezvoltare Z80 şi microcalculatorul integrat Z8. Multiplexorul este implementat cu 4 circuite 74LS157. Printr-un comutator se generează semnalul BUSRQ prin care se cere atribuirea memoriei RAM microcalculatorului integrat Z8/64. Microcalculatorul Z80 ca urmare a cererii BUSRQ, îşi trece magistralele în starea de impedanţă ridicată şi răspunde prin semnalul BUSAK. Acest semnal, BUSAK, intră pe pinul 1 (Select) din circuitele multiplexoare (74LS157) pen-



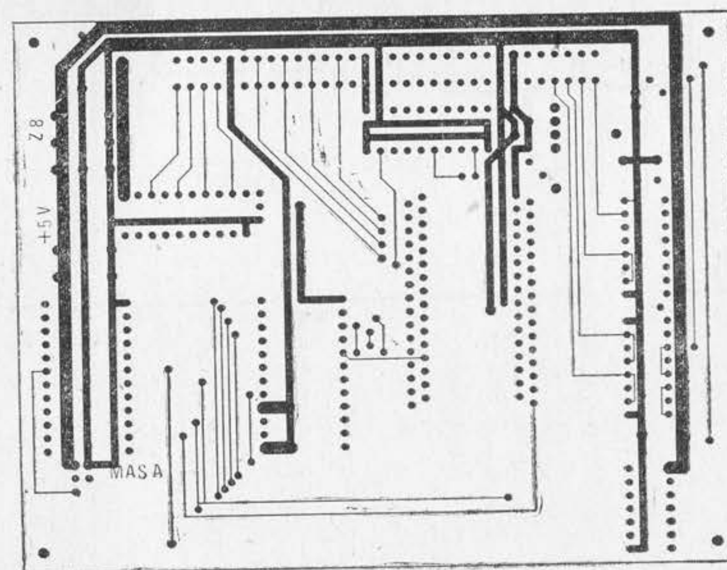


Fig. 3.4. Cablajul plăcuţei „Nucleu de dezvoltare cu Z8"

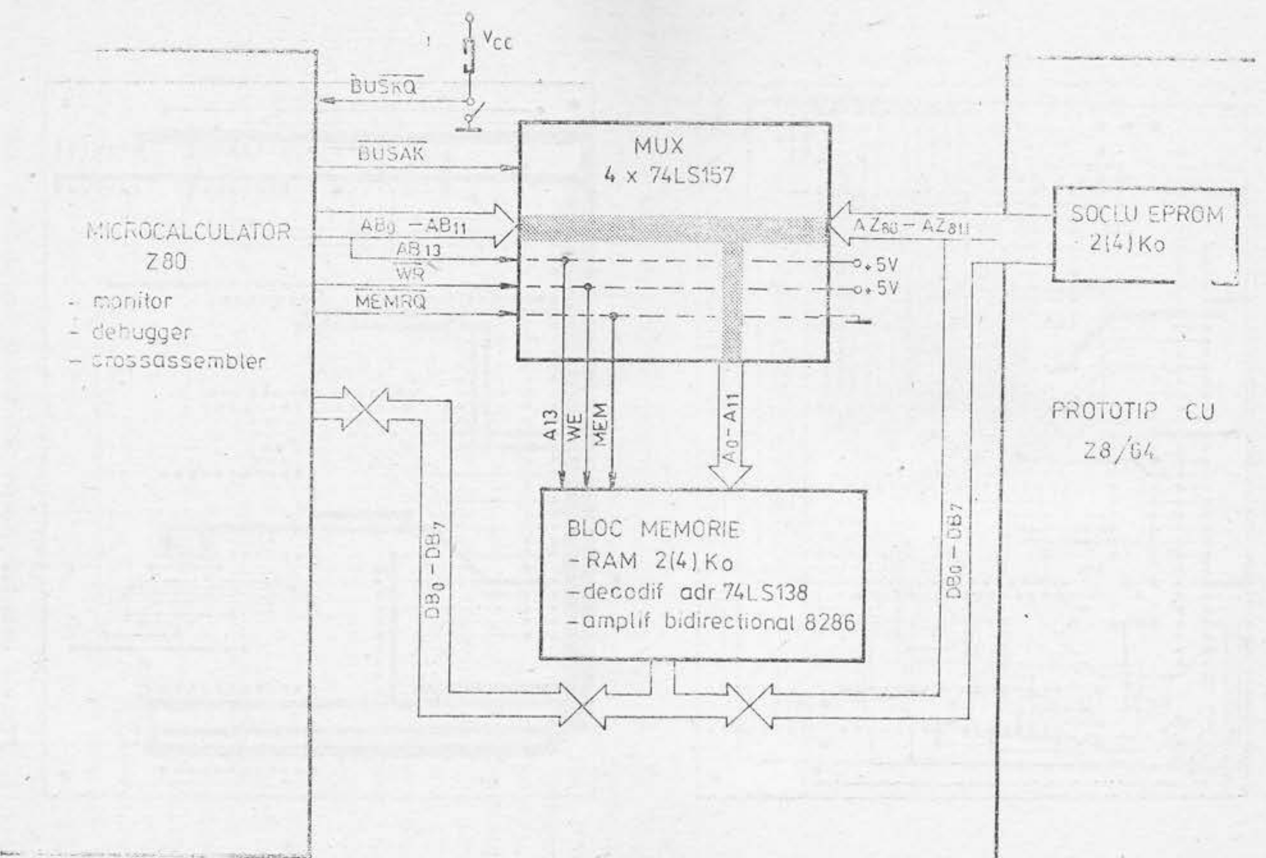


Fig. 3.5. Simulatorul de EPROM. Schema bloc

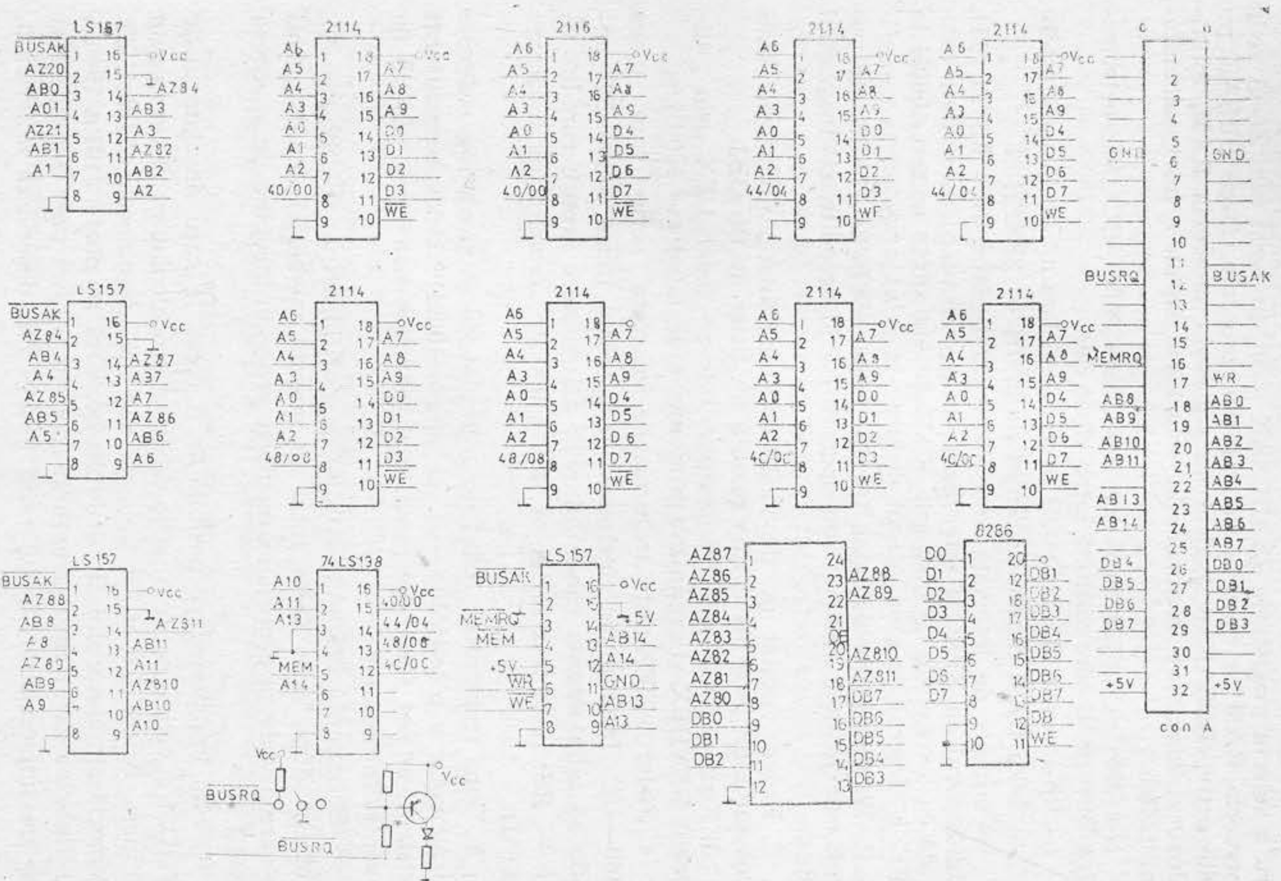


Fig. 3.6. Schema electrică

tru a selecta magistrala Z8 (BUSAK = 0). În acest caz, Z8-ul vede memoria RAM ca propria sa memorie program. Dacă BUSAK nu este activ (BUSAK = 1), deci nu s-a cerut memoria RAM, multiplexorul lasă spre memoria în cauză magistralele microcalculatorului Z80.

S-a ales ca microcalculator Z80 placa M80—UC (Microelectronica București) care dispune între altele de:

— un monitor de la C000H (o variantă lansată de firma Mostek)

— 16ko RAM dinamic aranjat de la 0H — 3FFFH și

— circuite de interfață serie pentru 2 legături full duplex [11].

Simulatorul de EPROM este văzut de această placă de unitate centrală ca o extensie de memorie RAM de 4ko de la adresa 4000H la 7FFFH. Lucrul cu simulatorul decurge în felul următor:

1. Pe un microcalculator (M118, P386, CUB Z, MADS etc) pe care este instalat sistemul de operare CP/M se editează programul Z8.

2. Fișierul editat de tip ASM este asamblat cu ajutorul unui crosasamblor, în urma căruia rezultă un fișier de tip OBJ.

3. Fișierul OBJ este transformat într-un fișier HEX prin utilitarul GENHEX și organizat în memorie de la adresa 4000H.

4. Fișierul HEX este transmis pe interfața serie spre placa M80—UC. Înainte de efectuarea transmisiei placa M80—UC trebuie să aibă asignat canalului logic OI: (obiect input), canalul fizic — RS: (interfața serie), și să fie lansată comanda de încărcare (LOAD).

5. După transmisie, codul programului Z8 se găsește în extensia de memorie a plăcii M80—UC la adresa 4000H. Aici sub monitorul existent se pot face alte modificări, adăugiri sau pune puncte de revenire în monitor.

6. Se comută butonul BUSRQ pe poziția activă (BUSRQ = 0), moment după care Z8/64 intră în posesia memoriei simulatorului pe care o vede ca propria sa memorie program începînd de la adresa 0000H.

O altă implementare posibilă ar fi aceea în care simulatorul de EPROM aparține pe de o parte direct microcalculatorului care poate prelucra fișiere CP/M, și pe care are loc asamblarea codului Z8. Memoria simulatorului fiind dublu accesată se poate utiliza fie un multiplexor ca în cazul prezentat mai sus, fie un port dublu. Acest gen de implementare evită pașii 3 și 4 din descrierea anterioară.

3.3. INSTRUMENTE SOFTWARE

Scopul instrumentelor software, în general, este acela de a facilita punerea la punct a programelor în faza de dezvoltare și de depanare a eventualelor defecte induse în fazele de exploatare. Datorită „ermetismului" microcalculatorului integrat Z8, semnalat încă de la început, a apărut nevoia de instrumente software care să permită:

— examinarea și modificarea registrelor interne și

— urmărirea execuției programelor.

Această cerință este mai acută decît în cazul microprocesoarelor cunoscute 8080, Z80, care permit rularea pas cu pas, (oprirea pe fiecare ciclu mașină) dar și rularea programelor din memoria citește/scrie cu posibilități de întreruperi (breakpoint) și trasare (DDT). Există mai multe instrumente care permit punerea la punct a programelor, ele fiind mai mult sau mai puțin eficiente. Iată enumerarea cîtorva din ele într-o scară a eficienței:

1. semafoare
2. monitoare
3. simulatoare
4. procedee mixte.

3.3.1. SEMAFOARE

Prin semafoare se poate indica faptul că în rulare programul a trecut prin anumite puncte ale sale; de asemenea se poate indica la nivelul semaforului ceva din context (de exemplu: valoarea indicatoarelor de stare). Din punct de vedere hard semaforul poate fi o diodă electroluminiscentă, un circuit de decodificare și afișare pe șapte segmente, o legătură serie cu o consolă etc. Dezavantajele acestui gen de instrumente sînt:

1. consumă resurse hard (linii paralele sau legătura serie)

2. secvențele program prin care se face semaforizarea constituie un apendice care consumă spațiu de memorie și timp. Dezavantajele menționate nu au importanță, în cazul aplicațiilor care nu consumă integral resursele hard disponibile ale microcalculatorului integrat și nu sînt critice în timp. Semaforizarea nu poate fi folosită în cazul aplicațiilor care au nevoie de toate resursele hard sau în care dimensiunea timp este critică.

3.3.2. MONITOARE

Monitoarele sînt programe intim legate atît de partea hardware a sistemului cît și de procesorul în sine. Ele sînt destinate controlului în anumite limite a stării mașinii și a desfășurării programelor. Comenzi de genul celor pentru lansarea unui program, vizualizarea sau modificarea unor registre, vizualizarea sau modificarea memoriei, încărcarea de programe și transmiterea de date sînt prezente în mai toate monitoarele cunoscute.

3.3.3. PROCEDEE MIXTE

Plecînd de la lipsa regimului pas cu pas pentru testarea programelor de dezvoltare s-au adoptat tehnici mixte. Astfel pentru testarea unor secvențe de program, se inserează încă din faza de editare a respectivei secvențe salturi de monitor. După lansarea secvenței program la trecerea prin aceste salturi, se revine în monitor unde se pot examina registrele interne și eventual se face modificarea conținutului acestora. Programul se reia printr-o comandă de execuție pe prima adresă după saltul în monitor. De asemenea s-a menținut acolo unde considerentele hardware au permis „semaforul".

3.4. MONITORUL DEP Z8

Pentru punerea la punct a software-ului microcalculatorului integrat Z8, autorul a pus la punct un program care permite vizualizarea regiștrilor interni, modificarea conținutului acestora, lansarea unui program de la o adresă dată și vizualizarea memoriei. Acest monitor a fost denumit depanator Z8, DEP Z8. Acesta face parte din resursele software minime necesare dezvoltării de produse cu ajutorul microcalculatorului integrat. Datorită faptului că magistrala de date D0 — D7, din cazul versiunii de dezvoltare Z8/64, este unidirecțională, nu s-au putut implementa comenzi de genul „breakpoint" cunoscute din cazul monitoarelor INTEL pentru microprocesorul 8080 sau MOSTEK pentru microprocesorul Z80.

ALOCAREA SPAȚIULUI DE MEMORIE

Lungimea monitorului este de aproximativ 1/2 ko. Monitorul DEP Z8 poate fi asamblat imediat după tabela vectorilor de întrerupere, deci de la adresa 000CH. Monitorul poate fi asamblat însă

și la capătul memoriei program, urmînd ca imediat după tabela vectorilor de întrerupere și subprogramul de configurare al microcalculatorului integrat Z8 să urmeze programul utilizator. Acest subprogram de configurare trebuie să fie comun aplicației și monitorului. La finele subprogramului de configurare va fi un salt în monitorul propriu-zis, în această a doua variantă de amplasare. Mai jos se prezintă cele două maniere de locare a monitorului sugerate.

ALOCAREA MONITORULUI

adr.	spațiul de memorie	
000	vectori de întrerupere	vectori de întrerupere
00C		
00D	subprogram configurare	subprogram configurare salt DEP Z8
	monitor DEP Z8	program utilizator
7FF (FFF)	program utilizator	monitor DEP Z8

CONCEPTELE MONITORULUI DEP Z8

Monitorul este gîndit ca făcînd parte dintr-un sistem om-mașină, care are la nivelul interfeței cu operatorul uman o consolă. Pentru această interfață s-a folosit circuitul de intrare/ieșire serie și un circuit de ceas împreună cu predivizorul său. Caracteristicile legăturii serie se stabilesc în conformitate cu caracteristicile consolei, în anumite limite, în cadrul subprogramului de configurare. Recepția, respectiv transmisia caracterelor se realizează utilizînd facilitățile de întrerupere. Astfel microcalculatorul integrat Z8 nu este blocat în așteptarea realizării comunicației efective: Caracterele recepționate se depun într-o zonă tampon din setul de registre, zonă numită BUD de 16 octeți lungime. Depunerea caracterelor se face în ordinea recepționării. Totodată se incrementează un indicator numit NRCBUD, care ține evidența numărului curent

al caracterelor din BUD. Depunerea caracterelor sosite pe recepția serie se face în ordinea sosirii la adresa imediat superioară celei precedente. Scoaterea caracterelor din BUD, în vederea analizei comenzii, se face în ordine, numai de la adresa de bază (BUD). După scoaterea unui caracter, celelalte se repliază în interiorul zonei tampon spre capul zonei. Altfel spus s-a implementat software o memorie FIFO. Caracterul recepționat este transmis înapoi (ecou), astfel că apariția comenzii, tastate la consola pe ecran, este și dovada recepționării ei. Monitorul constă dintr-o buclă principală de unde urmărește dacă sînt caractere recepționate în BUD. Monitorul se folosește de registrele interne din zona 60H — 7FH. Prin punerea stivei în registrele interne începînd cu registrul 5FH în loc de 7FH, practic am redus numărul registrelor de uz general care stăteau la dispoziția utilizatorului cu 32, dar am separat spațiul de lucru al monitorului. Dacă indicatorul NRCBUD este diferit de zero, se preia caracterul din BUD, repliind totodată restul caracterelor în bufer și decrementînd indicatorul NRCBUD. După care analizează caracterul în cauză, și dacă e mnemonica unei comenzi cunoscute se sare în subprogramul corespunzător de tratare a ei. Fiecare comandă cere 1 sau 2 parametri, pe care-i preia într-o manieră asemănătoare. Pentru modul său intim de lucru monitorul DEP Z8 are nevoie de niște locații de manevră și de niște indicatoare împreună cu zona tampon de recepție. Programele utilizatorului trebuie să evite utilizarea registrelor destinate monitorului. Pentru a resimți cît mai puțin această restricție aceste registre au fost alocate compact în zona de sfîrșit a setului de registre. În tabelul 3.1. este prezentat modul de alocare a setului de registre al monitorului.

COMENZILE MONITORULUI

Comenzile au următoarea formă generală:

[comanda] [operand 1] [separator] [operand 2] [terminator] unde:

— comanda — este mnemonica comenzii,

— operand 1 și 2 — este primul și al doilea operand,

— separator — caracterul 20H (spațiu) sau 27H (virgulă) și

— terminator — caracterul 0DH (retur de car).

De asemenea, o comandă se poate abandona dacă înainte de a introduce terminatorul se introduce caracterul 1BH (ESC-escape). Între mnemonica comenzii și primul operand monitorul va tipări în mod automat un spațiu. Operanzii vor avea maximum 2 sau 4 caractere; dacă s-au introdus mai multe caractere decît e nevoie se iau în considerare numai ultimele două sau patru, celelalte se neglijează. Comenzile recunoscute de monitorul DEP Z8 sînt:

1) E — execută program,

2) R — vizualizează registre,

3) — S — afișează și substituie conținutul registrelor, și

4) — M — vizualizează memoria program.

După punerea sub tensiune sau după acționarea butonului de inițializare (RESET) monitorul va tipări la consolă caracterul de

Tabelul 3.1. ALOCAREA REGISTRELOR

adresa	nume	setul de reg.	
7F	SPL	5F	adr. stivei
7E	BUD/BUF	FF/00	bufer DEPZ8/utilizator
7D	INDBUD	60 + NRCBUD	adr. ultimului car.rec.
7C	NRCBUD	×	
7B	MANL		păstrează rezultate
7A	MAN		parțiale
79	MANV		nesemnificative
78	INDR	adr. reg.	de afișat
77	AST/GA	FF/00	protecție SIO
76	INDP		
75	PAR2L		operand 2 low
74	PAR2		high
73	PAR1L		operand 1 low
72	PAR1		high
71	NROP	1/2	nr. operanzi
70	ACC		
6F	zona		
.	tampon		
60	BUD		
5F	baza stivă		
5E			
5D			
.			
	zona		
.	utilizator		
.			
04			
03	P3		
02	P2		
01	P1		
00	P0		

recunoaştere 24H ($), aşteptînd în continuare introducerea comenzilor. Cu ajutorul caracterului ESC se poate abandona o comandă în curs de livrare. De exemplu, forma următoare M 100, 37[ESC] nu se va mai executa, se iniţializează zona tampon, BUD, şi indicatorii caracterelor recepţionate, după care se sare din nou în bucla principală a monitorului republicîndu-se caracterul de recunoaştere.

Comanda E aaaaT — transferă controlul programului care se află la adresa aaaa. Exemple:

E 10<CR> — începe execuţia programului care se află în memoria program la adresa 0010H.

E 20010<CR> — are acelaşi efect cu comanda precedentă.

E D⟨CR⟩ — începe execuţia programului aflat la adresa 00DH

R aa,bbT — afişează pe rînduri de 16 caractere registrele dintre adresele aa şi bb; dacă bb nu are forma (aa + m*16) se afişează integral şi rîndul ultimului registru (bb). Exemple:

R ⟨CR⟩ — afişează primele 16 registre.

R 210⟨CR⟩ sau R 10, 17⟨CR⟩ — afişează 16 registre începînd cu adresa 10H.

R 23, 42⟨CR⟩ sau 23, 40⟨CR⟩ — afişează registrele începînd cu adresa 23H pînă la adresa 43H pe rînduri de 16 caractere.

Comanda *substitut* pune la dispoziţia utilizatorului nişte opţiuni; aceste opţiuni sînt prezentate între paranteze drepte; conţinutul registrului aa este redat în forma (aa).

De exemplu: S aa (aa)—[spaţiu] (aa + 1)— sau
S aa (aa)—[bb] (aa + 1)— ş.a.m.d.

Comanda poate fi continuată în următoarele moduri:

1) prin substituirea conţinutului registrului afişat după care se trece la următorul registru

2) prin trecerea la următorul registru fără modificarea celui curent prin introducerea caracterului 20H ⟨spaţiu⟩

3) prin revenirea la registrul precedent la introducerea caracterului 08H ⟨backspace⟩

4) prin terminarea ei cu ⟨CR⟩ şi

5) prin abandonarea cu ⟨ESC⟩ a ultimei modificări şi a comenzii.

Şi aici, în cazul că operatorul a introdus mai multe caractere hexa se iau în considerare numai ultimele două caractere.

Exemple:

Dacă primele 7 registre au valorile de mai jos:

17 20 30 21 EF 2B 07 C3

atunci după comanda:

S 00⟨CR⟩

dacă se derulează interactiv următoarea secvenţă:

17—[spaţiul] 20—[25] 30—[31] 21—[20] EF—[6] 2B—[27] 7—[07]
C3—[⟨CR⟩] conţinutul acestor prime 7 registre va fi:

17 25 31 20 06 27 7 C3

Trebuie remarcat că prin comanda S se dispune implicit şi de resursele de I/E prin accesul la conţinutul registrelor dedicate.

M aaaa, bbbb⟨CR⟩ — afişează memoria dintre adresele aaaa şi bbbb pe rînduri de 16 adrese. Toate celelalte observaţii cu privire la editarea comenzilor sînt valabile şi aici.

În figura 3.7. se prezintă organigrama monitorului.
Secvenţa de configurare începe la adresa 000CH. Aici se stabileşte între altele comunicaţia serie la 4800 biţi/sec, ocupînd în acest fel: ceasul T0 pentru tactul comunicaţiei serie, circuitul de intrare/ieşire serie şi pinii P30 şi P37 din portul P3. Tot în secvenţa de configurare se autorizează întreruperile pe recepţie (SI) şi transmisie (S0) serie. Caracterele recepţionate de la consolă de sistemul de dezvoltare cu Z8, care dispune de acest monitor, sînt depuse în buffer-ul BUD rezervat în zona 60H —6FH a registrelor interne. Recepţia datelor de la consolă se face automat; cu recepţia fiecărui caracter se incrementează registrul indicator al buffer-ului de recepţie (INDBUD) şi registrul numărului de caractere curent în buffer (NRCBUD). În secvenţa de configurare mai are loc:

1) stabilirea stivei interne la adresa 5FH

2) ştergerea cererilor anterioare de întrerupere

3) punerea măştilor corespunzătoare cererilor de întrerupere

4) fixarea lanţului de priorităţi

5) configurarea portului P3 pentru comunicaţie serie, a circuitului de ceas T0 şi a predivizorului acestuia PRE0 (configurarea porturilor P0, P1 şi P2, a circuitului de ceas T1 se face de acord cu necesităţile aplicaţiei preconizate a se dezvolta).

După configurare urmează o secvenţă de ştergere a registrelor interne şi o iniţializare a registrului INDBUD, care se continuă cu o secvenţă de afişare la consolă a unui caracter de recunoaştere, numit prompter, — 24H ($). Subrutinele de tratare a comenzilor se bazează pe un subset de subrutine de bază (primitive) care asigură

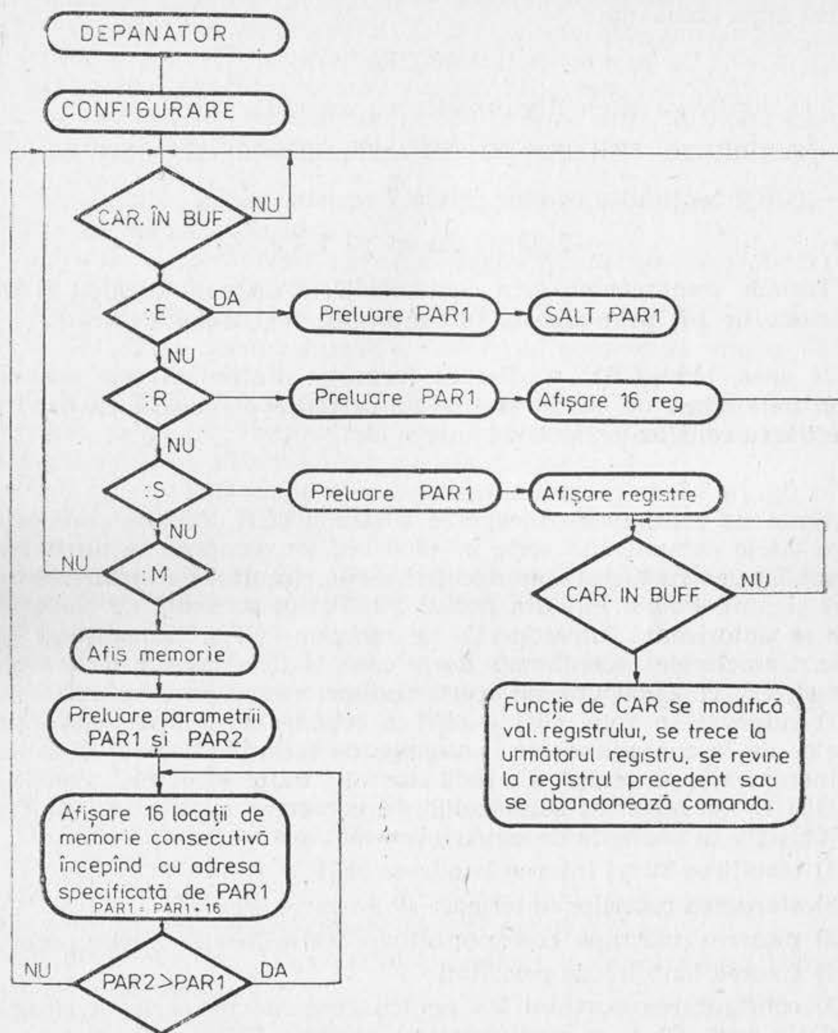


Fig. 3.7. Organigrama monitorului DEP Z8

operaţiile elementare de conversie ASCII/BINAR şi invers, afişarea acumulatorului, afişarea unui text, citirea operanzilor, deplasarea operanzilor şi reţinerea ultimelor patru cifre hexazecimale, filtrarea caracterelor etc.

SUBRUTINELE DE BAZĂ

Monitorul DEP Z8 cuprinde o serie de subrutine folositoare şi utilizatorului în dezvoltarea de programe proprii. Aceste subrutine pot fi apelate din monitor de către programul utilizator sau pot fi preluate şi incluse în programele utilizatorului. Între subrutinele monitorului care printr-un anumit grad de generalitate pot fi utile unui cerc mai mare de utilizatori sînt şi ASBIN, PACC, PRVAL, WRCHR, CRLF, RDOP. FILTRU.

Trebuie subliniat faptul că registrul botezat Acc nu trebuie înţeles în sensul noţiunii cunoscute în cadrul microprocesoarelor INTEL 8080 sau ZILOG Z80, ci este un registru oarecare a microcalculatorului integrat Z8 dedicat doar în monitor unei anumite funcţii.

1) ASBIN —subrutina de conversie caracter ASCII în semioctet hexa.

intrări: # Acc

ieşiri: # Acc (caracterul hexa codificat ASCII convertit binar)

2) PACC —subrutina de conversie în ASCII şi de afişare a conţinutului # Acc; apelează la subrutinele PRVAL şi WRCHR.

intrări: # Acc

ieşiri: —conţinutul # Acc transformat în două caractere ASCII transmise pe interfaţa serie, urmate de un caracter de spaţiu.

3) PRVAL —subrutina de conversie semioctet hexa în caracter ASCII

intrări: # Acc # (Acc AND 0FH)

ieşiri: caracterul ASCII, provenit din conversie, transmis pe interfaţa serie

4) WRCHR —subrutina de transmitere pe interfaţa serie a conţinutului # Acc

intrări: # Acc

ieşiri: conţinutul # Acc transmis pe interfaţa serie

Această subrutină asigură şi protecţie la suprascrierea registrului SIO.

5) CRLF —subrutina de transmitere pe interfaţa serie, două caractere ⟨CR⟩ şi ⟨LF⟩, pentru a asigura un rînd nou pe consolă

intrări: —

ieşiri: caracterele 0D H şi 0A H transmise pe interfaţa serie

6) RDOP —subrutina de preluare a 1 sau 2 operanzi

intrări: caracterele depuse în BUD şi # NROP

ieşiri: operanzii în forma binară depuşi în # PAR1, # PAR1L, # PAR2, # PAR2L

7) FILTRU —subrutina de semnalizare a unui caracter inafara celor admise

intrări: # Acc

ieşiri: inscrie CARRY pentru caracter invalid în # Acc, # Acc nemodificat.

În continuare listăm monitorul DEP Z8 asamblat cu ajutorul crosasamblorului CROSS (elaborat de Viorel Bălan de la IEMI Bucureşti). Acest utilitar, în varianta deţinută de autor, are o mică deficienţă şi anume adresele pe 2 octeţi sînt inversate. Adică sînt asamblate în maniera low, high ca la 8080 şi Z80.

Tabelul 3.2. MONITORUL DEP Z8

```
                     CROSS 80 v.2.0 PAGE 1
0000 :
0000 :
0000 :
0000 :               ; DEPANATOR Z8 pentru slave
0000 :               ; DISC 35 fata B
0000 :               ; 16.02.89/20.04/19.06
0000 :               ORG      0000H
0070 =               ACC:     EQU      70H
0071 =               NROP:    EQU      71H
0072 =               PAR1:    EQU      72H
0073 =               PAR1L:   EQU      73H
0074 =               PAR2:    EQU      74H
0075 =               PAR2L:   EQU      75H
0076 =               INDP:    EQU      76H
0077 =               ASTGA:   EQU      77H
0078 =               INDR:    EQU      78H
0079 =               MANV:    EQU      79H
007A =               MAN:     EQU      7AH
007B =               MANL:    EQU      7BH
000D =               INDBUF:  EQU      0DH      ; BUF Se referă la buf_
                                                  ferul de recepţie
0050 =               BUFF:    EQU      50H      ; al slave-ului
000C =               NRCBUF:  EQU      0CH
007D =               INDBUD:  EQU      7DH
0060 =               BUD:     EQU      60H
007C =               NRCBUD:  EQU      7CH
0000 :
0002 =               AFIS:    EQU      2
0000 :
0000 : 5B 02         DW       NREF            ; P32 — IRQ0
0002 : 5C 02         DW       CARRY           ; P33 — IRQ1
0004 : 5E 02         DW       BORROW          ; P31 — IRQ2
0006 : 1B 02         DW       SI              ; P30 — IRQ3
0008 : 48 02         DW       SO              ; T0 INTERN SAU S0
000A : 61 02         DW       T1              ; T1 INTERN
000C : 8B 00  TBSALT: JR      CONFIG
```


Tabelul 3.2. (continuare)

```
000E : E6 02 77 CONFIG LD    #AFIS,77H
0011 : 31 F0          SRP    0F0H
0013 : FC 4F          LD     R15,4FH       ; SPL
0015 : B0 FA          CLR    #0FAH         ; IRQ
0017 : BC 18          LD     R11, 18H      ; IMR = 10 : X : T1 : T0/S0 : S1 ;
                                             BORROW : CARRY : NREF :
0019 : 9C 2F          LD     R9, 2FH       ; IPR = 18H,   BORROW >
                                             > CARRY > T1 > SI >
                                             > NREF > S0
001B : 8C 45          LD     R8,45H        ; P01M, = P0-IN P1-OUT
                                             STIVA INT,
001D : 7C 43          LD     R7,43H        ; P3M = FĂRĂ  PARITATE,
                                             S1/S0,   RESTUL I/E
001F : B0 F6          CLR    #0F6H         ; P2M — OUT
0021 : 5C 05          LD     R5, 5         ; PRE0 = 1,    MODULO  N
0023 : 4C 0D          LD     R4,0DH        ; T0 = 13 — 1MHz/13    —
                                           — 4800 bit/sec
0025 : 3C A3          LD     R3,0A3H       ; PRE1 = 40,  TACT INTERN,
                                             MODULO N
0027 : 2C FA          LD     R2,0FAH       ; T1 = 250, 1MHz/250 * 40 =
                                             = 10ms
0029 : 1C 0F          LD     R1,0FH        ; ENABLE SI LOAD T0 SI T1
002B :          ;       DEPANATOR
002B : 31 00          SRP    0
002D : 4C 79          LD     R4,79H
002F : 5C 06          LD     R5,06
0031 : B1 05   STERS: CLR    C#5
0033 : 5E             INC    R5
00034 : 4A FB         DJNZ   R4, STERS
0036 : B0 05          CLR    #5
0038 : E6 7D 60       LD     #INDBUD,BUD   ; INITIALIZARE IND, BUD
                                               REC.
003B : 9F             EI
003C : E6 FE FF DEBUG : LD   #OFEH,OFFH    ; INDICATOR DEPANATOR
003F : 31 00          SRP    0
0041 : E6 70 0D       LD     #ACC,0DH
0044 : D6 41 01       CALL   WRCHR
                      CROSS-80 v.2.0. PAGE 2
0047 : E6 70 0A       LD     #ACC,0AH
004A : D6 41 01       CALL   WRCHR
004D : E6 70 24       LD     #ACC, 24H             ; AFIS. $
0050 : D6 41 01       CALL   WRCHR
0053 : 76 7C FF DEBUG1 : TM  #NRCBUD,      ; EX. CAR. IN BUF. REC. AL
                             OFFH            DEPANATOR ?
0056 : 6B FB          JR     Z,DEBUG1
0058 : E4 60 70       LD     #ACC,#BUD
005B : D6 C8 01       CALL   MOVBUD
005E : A6 70 45       CP     #ACC,45H
0061 : 6B 12          JR     Z,EXEC
0063 : A6 70 52       CP     #ACC,52H
0066 : 6B 12          JR     Z,REG
0068 : A6 70 53       CP     #ACC,53H
```

Tabelul 3.2. (continuare)

```
006B : 6B 21            JR     Z,SCH
006D : A6 70 4D         CP     #ACC,4DH
0070 : 6D E0 00         JP     Z,MEM
0073 : 8B C7            JR     DEBUG
0075 : D6 5B 01 EXEC:   CALL   RDOP
0078 : 30 72            JP     C#PAR1
007A :
007A :          ;       AFIŞARE 16 REGISTRE
007A :          ;       SPECIFICATE ÎN PARAM
007A :
007A : D6 5B 01 REG:    CALL   RDOP
007D : D6 4E 01         CALL   CRLF
0080 : BC 10            LD     R11,10H           ; FIXEZ NR DE REG DE AFI-
                                                   ŞAT
0082 : E5 73 70 REG1:   LD     #ACC,@#PAR1L     ; TREC PAR1 IN ACC
0085 : D6 1F 01         CALL   PACC              ; AFISEZ PAR
0088 : 20 73            INC    #PAR1L
008A : BA F6            DJNZ   R11,REG1
008C : 8B AE            JR     DEBUG
008E :          ;       SCHIMBA VALOAREA REGISTRELOR
008E :
008E : D6 5B 01 SCH:    CALL   RDOP              ; AST. PARAM.
0091 : E4 73 78         LD     #INDR,#PAR1L
0094 : 00 78            DEC    #INDR
0096 : 20 78    SCH1:   INC    #INDR
0098 : E6 70 20 SCH11:  LD     #ACC,20H                ; AFISEZ „SPATIU"
009B : D6 41 01         CALL   WRCHR
009E : E5 78 70         LD     #ACC,@#INDR      ; AFIS. CONTINUT REG.
00A1 : D6 1F 01         CALL   PACC
00A4 : E6 70 2D         LD     #ACC,2DH          ; AFISEZ —
00A7 : D6 41 01         CALL   WRCHR
00AA : 76 7C FF SCH2:   TM     #NRCBUD,0FFH     ; ASTEPT
00AD : 6B FB            JR     Z,SCH2            ; CARACTER DE PRECIZARE
                                                   A COM. „S"
00AF : E4 60 70         LD     #ACC,#BUD
00B2 : A6 70 20         CP     #ACC,20H
00B5 : 6B 24            JR     Z,SCH5            ; SPATIU ? — MERGI MAI
                                                   DEPARTE
00B7 : A6 70 0D         CP     #ACC,0DH
00BA : 65 1F            JR     Z,SCH5            ; CR ? — MERGI MAI DE-
                                                   PARTE
00BC : A6 70 5E         CP     #ACC,5EH
00BF : 6B 0D            JR     Z,SCH3            ; < ? — MERGI INAPOI
00C1 : A6 70 1B         CP     #ACC,1BH
00C4 : 6B 0F            JR     Z,SCH4            ; ESC- ? TERMINA COMANDA
00C6 : D6 5B 01         CALL   RDOP              ; PREIA VALOAREA NOUA
00C9 : F5 73 78         LD     @#INDR,           ; SCHIMBA VALOAREA
                               #PAR1L
00CC : 8B C8            JR     SCH1
00CE : 00 78    SCH3:   DEC    #INDR
00D0 : D6 C8 01         CALL   MOVBUD
00D3 : 8B C3            JR     SCH11
00D5 : D6 C8 01 SCH4:   CALL   MOVBUD
```


Tabelul 3.2. (continuare)

```
00D8 : 8D 3C 00         JP     DEBUG
00DB : D6 C8 01 SCH5:   CALL   MOVBUD
                               CROSS-80 v.2.0 PAGE 3
00DE : 8B B6            JR     SCH1
00E0 :
00E0 :                  ; AFISARE MEMORIE PROGRAM
00E0 :
00E0 : D6 5B 01 MEM:    CALL   RDOP              ; AST. ADR. LOW. HIGH
00E3 : D6 4E 01         CALL   CRLF
00E6 : BC 10    MEMO:   LD     R11, 10H
00E8 : 31 70    MEM1:   SRP    70H
00EA : C2 02            LDC    R0, CRR2
00EC : 31 00            SRP    0
00EE : D6 1F 01         CALL   PACC
00F1 : A0 72            INCW   #PAR1
00F3 : BA F3            DJNZ   R11, MEM1
00F5 : E6 70 0D         LD     #ACC, 0DH
00F8 : D6 41 01         CALL   WRCHR
00FB : E6 70 0A         LD     #ACC, 0AH
00FE : D6 41 01         CALL   WRCHR
0101 : A4 72 74         CP     #PAR2, #PAR1
0104 : BB E0            JR     UGT, MEMO
0106 : 6B 03            JR     EQ, MEM2
0108 : 8D 3C 00         JP     DEBUG
010B : A4 73 75 MEM2:   CP     #PAR2L, #PAR1L
010E : BB D6            JR     UGT, MEMO
0110 : 8D 3C 00         JP     DEBUG
0113 :
0113 :          ;       ASBIN — CONVERSIE CARACTER ASCII — ÎN
                          SEMIOCTET HEXA
0113 :          ;       INTRARE — Acc
0113 :          ;       IESIRE — Acc
0113 :
0113 : 26 70 30 ASBIN:  SUB    #ACC, 30H
0116 : A6 70 0A         CP     #ACC, 0AH
0119 : 7B 03            JR     ULT, ASBIN1
011B : 26 70 07         SUB    #ACC, 7
011E : AF       ASBIN1: RET
011F :
011F :          ;       PACC — AFISEAZĂ CONTINUT ACC
011F :          ;       INTRARE — Acc
011F :          ;       IESIRE — Două caractere ASCII pe interfaţa serie
011F :
011F : 70 70    PACC:   PUSH   #ACC
0121 : F0 70            SWAP   #ACC
0123 : D6 32 01         CALL   PRVAL             ; AFISEAZA SEMIOCTET SUPE-
                                                   RIOR
0126 : 50 70            POP    #ACC
0128 : D6 32 01         CALL   PRVAL             ; AFISEAZA SEMIOCTET INFE-
                                                   RIOR
012B : E6 70 20         LD     #ACC,20H          ; AFIS. SPATIU
```

Tabelul 3.2. (continuare)

```
012E: D6 41 01           CALL   WRCHR
0131: AF                 RET
0132:
0132:           ;        PRVAL – CONVERSIE SEMIOCTET HEXA IN ASCII
0132:           ;        INTRARE – Acc AND OFH
0132:           ;        IESIRE – INTERFATA SERIE
0132:
0132: 56 70 0F  PRVAL:   AND    #ACC, 0FH
0135: 06 70 90           ADD    #ACC, 90H
0138: 40 70              DA     #ACC
013A: 16 70 40           ADC    #ACC, 40H
013D: 40 70              DA     #ACC
013F: 8B 00              JR     WRCHR
0141:
0141:           ;        WRCHR – SCOATE PE INTERFATA SERIE Acc
0141:
0141: E6 77 FF  WRCHR:   LD     #ASTGA, 0FFH   ; INC. REG. AST/GATA
0144: E4 70 F0           LD     SIO,#ACC
                 CROSS 80 v.2.0 PAGE 4
0147: 9F                 EI
0148: 76 77 FF  WRCHR1:  TM     #ASTA,0FFH     ; ASTEPT 00 IN REG. 77H
014B: EB FB              JR     NZ, WRCHR1
014D: AF                 RET
014E:
014E:           ;        CRLF – RIND NOU PE CONSOLA
014E:
014E: E6 70 0D  CRLF:    LD     #ACC, 0DH
0151: D6 41 01           CALL   WRCHR
0154: E6 70 0A           LD     #ACC, 0AH
0157: D6 41 01           CALL   WRCHR
015A: AF                 RET
015B:
015B:           ;        CITIRE – 1 SAU 2 OPERANZI DIN „BUF REC"
015B:           ;        TERMINATOR – „CR".
015B:           ;        SEPARATOR – „,"
015B:           ;        ABANDON – „ESC"
015B: E6 7B 70  RDOP:    LD     #MANL, 70H     ; 70H = INCEPUT ZONA DE
                                                 STERS
015E: BC 07              LD     R11, 7         ; 7 OCTETI DE STERS
0160: B1 7B     RDOP1:   CLR    @#MANL
0162: 20 7B              INC    #MANL
0164: BA FA              DJNZ   R11, RDOP1
0166: E6 76 73           LD     #INDP,PAR1L    ; INDP = LOC. PARAM.1
0169: E6 71 01           LD     #NROP,01H      ; ASTEPT PRIMUL PARAM.
016C: 76 7C FF  RDOP2:   TM     #NRCBUD, 0FFH
016F: 6B FB              JR     Z, RDOP2
0171: E4 60 70           LD     #ACC,#BUD
0174: D6 C8 01           CALL   MOVBUD
0177: A6 70 0D           CP     #ACC, 0DH      ; TERMINATOR ?
017A: 6B 2C              JR     Z, RETRD
017C: A6 70 20           CP     #ACC, 20H
017F: 6B 27              JR     Z, RETRD
```



```
0181: D6 FE 01           CALL   FILTRU
0184: 7B 22              JR     C, RETRD       ; NU A FOST CARACTER
                                                 VALID
0186: A6 70 2C           CP     #ACC, 2CH      ; SEPARATOR ?
0189: 6B 11              JR     Z, RDOP3
018B: D6 A9 01           CALL   DEPL           ; DEPL. INDP PT NOUL
                                                 SEMIOCTET
018E: D6 13 01           CALL   ASBIN
0191: 56 70 0F           AND    #ACC, 0FH
0194: 45 76 70           OR     #ACC, @#INDP
0197: F5 70 76           LD     @#INDP, #ACC
019A: 8D D0              JR     RDOP2
019C: 76 71 01  RDOP3:   TM     #NROP, 02H
019F: EB 07              JR     NZ, RETRD
01A1: E6 76 75           LD     #INDP, PAR2L   ; INDP = LOC. PARAM. 2
01A4: 20 71              INC    #NROP
01A6: 8B C4              JR     RDOP2
01A8: AF        RETRD:   RER
01A9:
01A9:
01A9:           ;        DEPL – DEPLASARE
01A9:           ;        SINT RETINUTE ULTIMELE 4 CIFRE ALE UNUI OPE-
                           RAND
01A9:           ;        DEPL SE FACE ASUPRA PARAMETRULUI SPECI-
                           FICAT DE INDP
01A9:
01A9: F1 76     DEPL:    SWAP   @#INDP
01AB: 71 76              PUSH   @#INDP
01AD: 57 76 0F           AND    @#INDP, 0FH
01B0: E5 76 7B           LD     #MANL, @#INDP
01B3: 51 76              POP    @#INDP
01B5: 57 76 F0           AND    @#INDP, 0F0H
01B8: 00 76              DEC    #INDP
01BA: F1 76              SWAP   @#INDP
01BC: 57 76 F0           AND    @#INDP, 0F0H
                 CROSS 80 v.2.0 PAGE 5
01BF: 05 76 7B           ADD    #MANL, @#INDP
01C2: F5 7B 76           LD     @#INDP, #MANL
01C5: 20 76              INC    #INDP
01C7: AF                 RET
01C8:
01C8:           ;        MOVBUD – DEPL. LA STINGA CU UN CARACTER
                           CARACTERELE DIN BUF. REC.
01C8:
01C8: 8F        MOVBUD:  DI
01C9: B8 7C              LD     R11, #NRCBUD
01CB: E6 7A 60           LD     #MAN, BUD
01CE: E6 7B 61           LD     #MANL, BUD+1
01D1: E5 7B 79  REPET:   LD     @#MANV, @#MANL
01D4: F5 79 7A           LD     #MAN, #MANV
01D7: 20 7A              INC    #MAN
01D9: 20 7B              INC    #MANL
```

Tabelul 3.2. (continuare)

```
01DB : BA F4            DJNZ   R11, REPET
01DD : 00 7D            DEC    #INDBUD       ; ACT. PRIMEI LOC. DIN BUF.
                                               LIBERE
01DF : 00 7C            DEC    #NRCBUD
01E1 : 9F               EI
01E2 : AF               RET
01E3 :
01E3 :            ;     MOVBUF — Deplasare la stg. cu 1 caracter a Bufferului
                          Depanatorului
01E3 : 8F       MOVBUF: DI
01E4 : B8 0C            LD     R11, #NRCBUF
01E6 : E6 7A 50         LD     #MAN, BUFF
01E9 : E6 7B 51         LD     #MANL, BUFF+1
01EC : E5 7B 79  RELUA: LD     #MANV, @#MANL
01EF : F5 79 7A         LD     @#MAN, #MANV
01F2 : 20 7A            INC    #MAN
01F4 : 20 7B            INC    #MANL
01F6 : BA F4            DJNZ   R11, RELUA
01F8 : 00 0D            DEC    #INDBUF
01FA : 00 0C            DEC    #NRCBUF
01FC : 9F               EI
01FD : AF               RET
01FE :
01FE :            ;     FILTRU LASA SA TREACA CARACTER HEXA ŞI ,,?''
01FE :            ;     NU DISTRUGE ACC
01FE :            ;     INSCRIE CARRY DACA ACC ARE CARACTER INVALID
01FE : A6 70 2C FILTRU: CP     #ACC, 2CH
0201 : 6B 14            JR     Z, OK
0203 : A6 70 30         CP     #ACC, 30H
0206 : 7B 11            JR     C, NOK
0208 : A6 70 3A         CP     #ACC, 3AH
020B : 7B 0A            JR     C, OK
020D : A6 70 41         CP     #ACC, 41H
0210 : 7B 07            JR     C, NOK
0212 : A6 70 47         CP     #ACC, 47H
0215 : FB 02            JR     NC, NOK
0217 : CF        OK     RCF
0218 : AF               RET
0219 : DF        NOK:   SCF
021A : AF               RET
021B :
021B :            ;     TRATARE INTRERUPERI
021B :            ; CARACTERELE RECEPTIONATE SE DEPUN IN BUF. REC ŞI
                    SE TRANSMIT
021B :            ; IN ECOU. ,,CR'' NU SE TRANSMITE IN ECOU
021B : D6 4C 02  SI:    CALL   TIMP
021E : 76 FE FF         TM     #0FEH, 0FFH
0221 : EB 16            JR     NZ, SID
0223 : F5 F0 0D         LD     @#INDBUF, SIO   ; PREIAU CARACTERUL
0226 : 57 0D 7F         AND    @#INDBUF,7FH    ; ECRANEZ BITUL DE PA-
                                                 RITATE
0229 : E5 0D 70         LD     #ACC, @#INDBUF  ; ECOU
```


Tabelul 3.2. (continuare)

```
                        CROSS-80 v.2.0 PAGE 6
022C : 20 0D            INC    #INDBUF
022E : 20 0C            INC    #NRCBUF
0230 : A6 70 0D  SI0:   CP     #ACC, 0DH     ; DACĂ E TERMINATOR
                                               ATUNCI
0233 : 6B 03            JR     EQ, SI1       ; NU TRANSMITE ECOU
0235 : D6 41 01         CALL   WRCHR         ; ECOU
0238 : BF        SI1:   IRET
0239 : F5 F0 7D  SID:   LD     @#INDBUD, SIO
023C : 57 7D 7F         AND    @#INDBUD, 7FH
023F : E5 7D 70         LD     #ACC, @#INDBUD
0242 : 20 7D            INC    #INDBUD
0244 : 20 7C            INC    #NRCBUD
0246 : 8B E8            JR     SI0
0248 :
0248 : E6 77 00  SO:    LD     #ASTGA, 0
024B : BF               IRET
024C : 31 00     TIMP:  SRP    0
024E : BC FF            LD     R11, 0FFH
0250 : 70 0B     TIMP1: PUSH   #0BH
0252 : BC 7F            LD     R11, 7FH
0254 : BA FE     TIMP2: DJNZ   R11, TIMP2
0256 : 50 0B            POP    #0BH
0258 : BA F6            DJNZ,  R11, TIMP1
025A : AF               RET
025B :                  ;
025B :                  ; Rutina tratare a intreruperii de la limitatori
025B :                  ;
025B : BF        NREF:  IRET
025C :                  ; Rutina de tratare a intreruperii de la CARRY
025C :                  ;
025C : 4E        CARRY: INC    R4
025D : BF               IRET
025E :                  ;
025E :                  ; Rutina de tratare a intreruperii de la BORROW
025E :                  ;
025E : 00 E4     BORROW: DEC   R4
0260 : BF               IRET
0261 :                  ;
0261 : BF        T1:    IRET
0262 :                  ;
0262 :                  ; Tratare eroare
0262 :                  ;
0262 :                  ;
0262 :                  END
                        CROSS—80 v.2.0 PAGE S

Symbols:
0070  ACC     0002 AFIS    0113 ASBIN   011E ASBIN1 0077 ASTGA
025E  BORROW  0060 BUD     0050 BUFF    025C CARRY  000E CONFIG
014E  CRLF    003C DEBUG   0053 DEBUG1  01A9 DEPL   0075 EXEC
01FE  FILTRU  007D INDBUD  000D INDBUF  0076 INDP   0078 INDR
```

Tabelul 3.2. (continuare)

007A	MAN	007B	MANL	0079	MANV	00E0	MEM	00E6	MEM0
00E8	MEM1	010B	MEM2	01C8	MOVBUD	01E3	MOVBUF	0219	NOK
007C	NRCBUD	000C	NRCBUF	025B	NREF	0071	NROP	0217	OK
001F	PACC	0072	PAR1	0073	PAR1L	0074	PAR2	0075	PAR2L
0132	PRVAL	015B	RDOP	0160	RDOP1	016C	RDOP2	019C	RDOP3
007A	REG	0082	REG1	01EC	RELUA	01D1	REPET	01A8	RETRD
008E	SCH	0096	SCH1	0098	SCH11	00AA	SCH2	00CE	SCH3
00D5	SCH4	00DB	SCH5	021B	SI	0230	SIO	0238	SI1
0239	SID	0248	SO	0031	STERS	0261	T1	000C	TBSALT
024C	TIMP	0250	TIMP1	0254	TIMP2	0141	WRCHR	0148	WRCHR1

No fatal errors

3.5. APLICAȚII

Plecînd de la nucleul de dezvoltare prezentat s-a dezvoltat o placă de încercare care urmărea în final punerea la punct a unui modul de control a poziției pe un grad de libertate la un robot industrial (v. fig. 3.8.). Schema bloc a primei faze a acestei aplicații este redată în figura 3.8. Pe această aplicație ne-am însușit și completat datele despre noul produs microelectronic, prin mici programe care au fost ușor de scris și urmărit. Tot pentru această fază de început s-a implementat o formă de „semafor" prin care se putea indica locul pe unde a trecut programul. Pe portul P2 s-a pus un decodificator/driver 7 segmente (MC 4511) și două afișoare de 7 segmente tip MDE 2102. În această fază Z8-ul era o cutie neagră în cea mai concretă accepțiune a sintagmei. După ce s-au pus la punct mici programe prin care s-au validat, corectat și completat informațiile oferite de [1] și [2] am trecut la scrierea și punerea la punct a monitorului DEP Z8. Numai cu ajutorul acestuia s-a putut dezvolta modul de control a mișcării. În continuare, ne vom opri pe scurt la două aplicații de generare de semnale.

3.5.1. GENERATOARE DE FUNCȚII

Folosindu-ne de convertorul numeric/analogic de 8 biți, DAC 08, montat pe portul P1, am dezvoltat două mici programe prin care am generat semnale periodice de diferite amplitudini și frecvențe. Primul subprogram schimbă frecvența și amplitudinea după epuizarea unor iterații programate în bucle. În cel de al doilea program schimbarea frecvenței se face cu ajutorul întreruperilor. Întreruperile sînt programate pe circuitul de ceas T1. Formele de un-

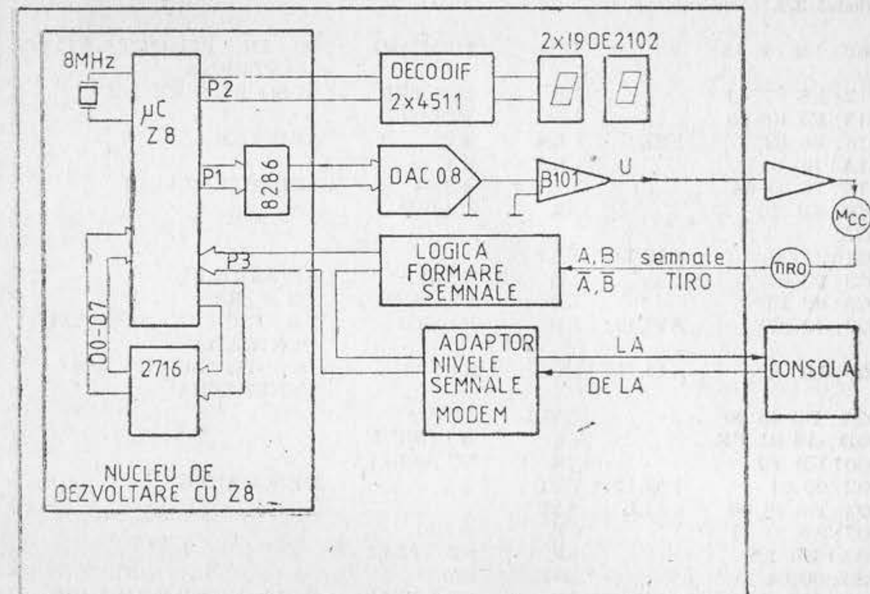


Fig. 3.8. Modul de control a poziției

dă generate se pot vizualiza cu un osciloscop pe ieșirea circuitului β101. Programul Generator de funcții este prezentat mai jos în tabelul 3.3. Alegerea unui program sau a celuilalt se face testînd o linie de intrare, P33, pe care s-a montat un comutator care oferă 0/5V. Faptul că este într-un program sau altul este redat pe afișoarele menționate prin cifrele 01 și 02.

Tabelul 3.3.

```
                    CROSS–80 v.2.0 PAGE 1
0000:
0000:
0000:                ; GENERATOR FUNCTII PE NUCLEU Z8 + DEZV. 1
0000:                ; LUCRAT LA 47 HZ
0000:                ; 12.08.88/16.01.89
0000:                ; DISC 35 FATA A
0000:
0000:                         ORG 10
000A:
000A: 66 00                   DW      T1
000C E6 FF 70    CONFIG: LD           SPL,70H
```

Tabelul 3.3. (continuare)

```
000F : E6 F8 45            LD      P01M,45H     ; P0—IN, P1—OUT, STIVA—
                                                 —INTERNA
0012 ; E6 F7 43            LD      P3M,43H      ; SI,SO RESTUL I/E
0015 ; E6 F6 00            LD      P2M,00
0018 : B0 02      SEL :    CLR     #2           ; AFISARE
001A : B0 01               CLR     #1
001C : 76 03 04            TM      #3,04        ; SELECTEAZA
001F : 6B 2B               JR      Z,APL2
0021 :
0021 : 31 00      APL1 :   SRP     0
0023 : 2C 01               LD      R2,01        ; AFISARE 01
0025 : 6C FF               LD      R6,OFFH      ; T/2 = (R5)
0027 : 4C 3F      APL10 :  LD      R4,3FH       ; NR. CIC. CU ACEEASI
                                                  PERIOADA
0029 : 1E         APL11 :  INC     R1           ; CRESTE CU 10V/256
                                                  TENSIUNEA
002A : D6 46 00            CALL    AST
002D : A6 01 FF            CP      #1,OFFH
0030 : EB F7               JR      NZ,APL11
0032 : 00 01      PAL12 :  DEC     #1           ; DESCRESTE
0034 : D6 46 00   CALL     AST
0037 : A6 01 01            CP      #1,01
003A : EB F6               JR      NZ,APL12
003C : 00 04               DEC     #4
003E : EB E9               JR      NZ,APL11     ; REIA DE (R4) ORI CIC. CU
                                                  ACEIASI PERIOADA
0040 : 00 06               DEC     #6           ; SCADE PERIOADA NOULUI
                                                  TREN DE IMPULSURI
0042 : EB E3               JR      NZ,APL10
0044 : 8B D2               JR      SEL
0046 :
0046 : 6C 77      AST :    LD      R6,77H
0048 : FF         AST1 :   NOP
0049 : 6A FD               DJNZ    R6,AST1
004B : AF                  RET
004C :
004C :            ; GENERATOR DE SEMNAL
004C :            ; PE BAZA DE INTRERUPERI
004C :
004C : 31 00      APL2 :   SRP     0
004E : 2C 02               LD      R2,02        ; AFISARE
0050 : 5C 7F               LD      R5,7FH       ; INDICATOR CRESTERE/
                                                  DESCRESTERE
0052 : E6 F3 1F            LD      PRE1,1FH     ; PRE1 = 7, MODULO N,
                                                  TIN=INTERN
0055 : E6 F1 8C            LD      TMR,8CH      ; TOUT=T1, LOAD T1,
                                                  ENABLE T1
0058 : E6 F2 FF            LD      T1,OFFH
005B : E6 FB E0            LD      IMR,0E0H     ; VALIDARE INT. T1
005E : E6 F9 CF            LD      IPR,0CFH     ; IRQ5(T1)>
0061 : 1C FF               LD      R1,0FFH      ; PALIER POZITIV
0063 : FF         APL21 :  NOP
```


Tabelul 3.3. (continuare)

```
0064 : 8B FD               JR      APL21
0066 :
0066 : 52 55      T1 :     AND R5,R5            ; DACA E FFH DESCRESTE
0068 : 6B 09               JR      Z,T12
006A : 00 F2               DEC     T1           ; DECREMENTEAZA VALOA-
                                                  REA CEASULUI T1
006C : 6B 03               JR      Z,T11        ; DACA T1 A ATINS O INCEPE
                                                  SA CREASCA
006E : 60 01      T10 :    COM     #1           ; INVERSEAZA PALIERUL
0070 : BF                  IRET
                  CROSS—80 v.2.0 PAGE 2
0071 : 60 05      T11 :    COM     #5           ; CRESTE PERIOADA
0073 : 20 F2      T12 :    INC     T1
0075 : A6 F2 7F            CP      T1,7FH
0078 : 6B F7               JR      Z,T11        ; PERIOADA A ATINS MAX
                                                  SALT LA DESCRESTERE
007A : 8B F2               JR      T10
007C :
007C :                     END
```

Aceste programe foarte scurte au și meritul de a arăta „puterea acestor noi membri ai familiei microelectronicii, de a arăta cît de ușor poți modela o aplicație dată. Exact această mică și neînsemnată aplicație prezentată mai sus poate fi adaptată doar prin program pentru a deveni baza unui aparat medical de recuperare prin impulsuri electrice. Aceste mici aplicații arată mai ales cu ce „resurse“ materiale derizorii se poate realiza astăzi ceea ce ieri presupunea un sertar sau un dulap de cîteva kilograme.

Esențial este să se înțeleagă faptul că prin adaugare de „inteligență“ partea materială se reduce continuu. Esențial mai este faptul că, forțînd puțin lucrurile, putem spune că suportul material al inteligenței poate fi oricît de mic.

Aceste considerente explică miniaturizarea din microelectronică și succesul ei și al informaticii acum și în viitorul previzibil.

4. MICROCALCULATOARE INTEGRATE SPECIALIZATE. TRANSPUTERE

4.1. PRIVIRE DE ANSAMBLU

Transputerul este o componentă VLSI programabilă. Este un microcalculator integrat specializat care are propria sa memorie locală și circuite de legătură pentru conectarea punct la punct cu alte transputere. Conectarea punct la punct se face printr-un set de legături (canale) seriale după un protocol precizat și cu frecvența fixă. Comunicarea punct la punct, unul din conceptele de bază ale transputerului, are următoarele avantaje asupra magistralei multiprocesor:

— nu există o limitare a numărului de transputere conectate datorate sarcinii capacitive,

— lărgimea de bandă a comunicației este suficientă, nu se saturează odată cu creșterea numărului de transputere din sistem. Esența arhitecturii unui transputer este prezentată în figura 4.1. Trebuie precizat de la început că prin transputer nu înțelegem un circuit ci o familie de circuite de diferite performanțe dar care, toate, respectă arhitectura și modul de comunicație prezentat. Transputerul poate fi utilizat într-un sistem cu un singur procesor sau într-o rețea. Printr-o rețea de transputere se urmărește construirea de sisteme performante în ceea ce privește concurența. Se pot construi cu el atît mașini SIMD, dar sînt mai ales avute în vedere pentru mașinile MIMD.

Transputerele pot fi programate în limbaje de nivel înalt, cum sînt limbajele C, Pascal, Fortran. În cazul rețelelor de transputere, unde trebuie exploatată concurența, programarea trebuie făcută într-un limbaj propriu. Pentru transputerele firmei INMOS, la care ne referim în mod special, acest limbaj propriu se numește OCCAM. Acesta este un limbaj de nivel înalt care asigură o eficiență maximă prin faptul că este croit special pentru transputere și conceput odată cu ele. OCCAM asigură utilizarea optimă a facilităților transputerului de la configurarea sistemului pînă la lucru în timp real, tratarea întreruperilor și a operațiilor de intrare/ieșire. Sarcina proiectantului de sistem este ușurată de relația dintre OCCAM și transputer. În OCCAM se definește noțiunea extrem de largă de „proces". Procesul putînd fi privit ca o cutie neagră, cu stări interne, care comunică cu alte procese prin mesaje. Un program rulînd într-un transputer este echivalent cu un proces OCCAM, astfel o rețea de transputere poate fi descrisă ca un proces OCCAM. Procesele pot să reprezinte o poartă logică, un microprocesor, un birou de aprovizionare, un sistem ecologic, un șef de compartiment etc. Un proces se poate compune din alte procese. Procesul este finit. Fiecare începe, dezvoltă o serie de acțiuni după care se termină. Acțiunile pot fi un set de procese secvențiale (programarea clasică) sau un set de procese paralele (care decurg în același timp). Deoarece un proces poate fi compus din alte procese rezultă că pot coexista acțiuni secvențiale și paralele. Procesele elementare, primitivele, sînt asignarea, intrarea și ieșirea. Procesele pot fi implementate în hard, în software și firește, într-un sistem hard-soft.

Dezvoltarea unui sistem constă în interconectarea de procese. Fiecare proces poate fi privit ca o unitate independentă care comunică cu alte procese prin canalele punct la punct. Comunicarea

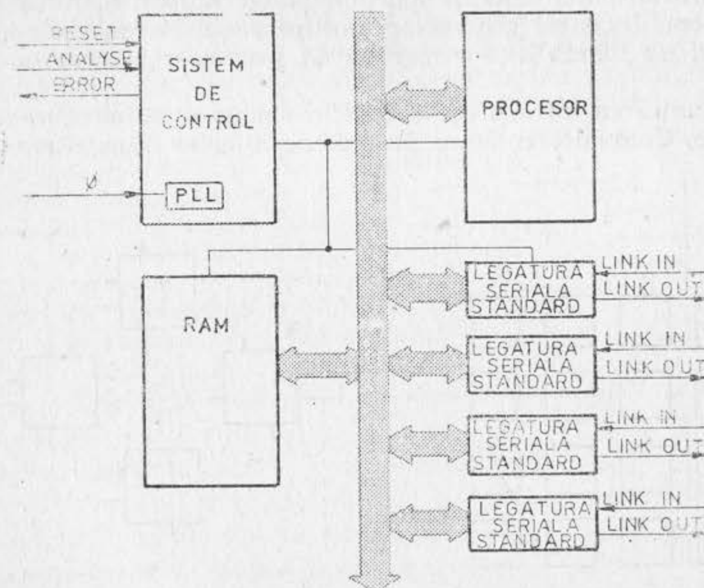


Fig. 4.1. Arhitectura generală a unui transputer

este sincronizată, nefiind necesar un alt mecanism de sincronizare. Aceste canale de comunicație punct la punct sincronizate le vom numi în continuare legături standard complet definite prin mesajele trimise în exterior. Din aceste motive putem considera că un sistem este structurat pe ierarhii. La orice nivel proiectantul se ocupă de un set de procese mici și controlabile, de complexitate redusă care permit stăpînirea și înțelegerea lor.

4.2. ARHITECTURA SISTEMELOR

Fiecare membru al familiei de transputere are unul sau mai multe legături seriale standard cu care poate fi conectat cu alte componente din familie. Această caracteristică permite construirea de rețele cu o topologie și un număr de componente impuse de o aplicație dată. În figura 4.2. se prezintă o rețea de transputere împreună cu detaliile unui nod de rețea. Fiecare transputer din sistem folosește propria sa memorie locală. Deoarece interfața cu memoria nu este disputată și de alte procesoare și este separată de interfața de comunicare, memoria locală poate fi optimizată funcție de aplicația în cauză. Lărgimea de bandă a tuturor memoriilor sistemului multitransputer este proporțională cu numărul transputerelor din sistem. În cazul sistemelor multiprocesor cu memorie comună lărgimea de bandă este împărțită de procesoarele care au acces la ea.

Comunicarea între procese într-un singur transputer se face via memorie. Comunicarea între procese pe diferite transputere se face

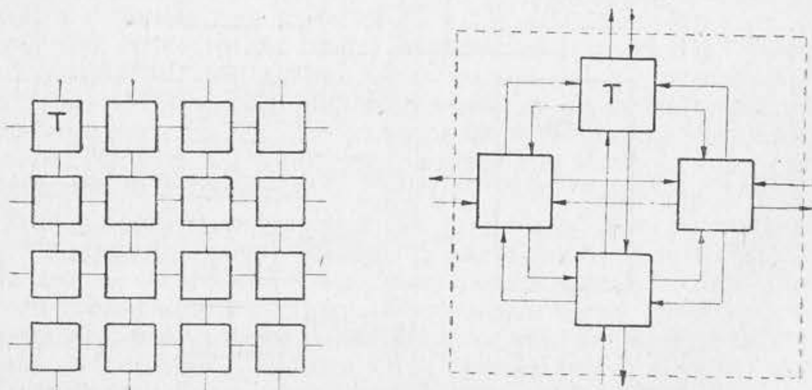


Fig. 4.2. Moduri de conectare a transputerelor

via legătura standard. Sincronizarea comunicației presupune însoțirea ei de mesaje suplimentare de recunoaștere. Prin urmare, o legătură presupune două linii, una pentru fiecare direcție. Pe aceste linii datele sînt transmise serial. Fiecare linie e folosită atît pentru date cît și pentru control. Protocolul de comunicație permite transmiterea unei secvențe arbitrare de octeți, fapt ce asigură conectarea de transputere avînd diferite lungimi de cuvînt. Fiecare mesaj este transmis ca o secvență de octeți, necesitînd un bufer de recepție de numai un octet pentru a fi siguri că nu se pierde informația. Fiecare octet transmis are forma:

start	1	B0	B1	B2	B3	B4	B5	B6	B7	stop

Cel care a transmis așteaptă un mesaj de recunoaștere din partea celui care a recepționat. Forma mesajului de recunoaștere are forma:

start	0

Bitul de start are valoarea 1, iar cel de stop are valoarea 0. Recunoașterea înseamnă două lucruri: 1) procesul a fost capabil să recepționeze și 2) că se poate transmite următorul caracter. Mesajul de recunoaștere se poate transmite imediat ce a început recepția octetului (după ce a sosit bitul de start). Ca urmare, transmisia poate fi continuă, fără întîrzieri între octeții de date. Toate transputerele au o frecvență standard de comunicație de 10 Mbiți/s, indiferent de performanțele procesorului. Astfel, transputere de diferite performanțe, de diferite generații pot fi direct interconectate, și sistemele de mîine vor comunica cu cele de azi. Legătura de comunicație nu este afectată de faza tactului (clock in) cerut de transputer. Acest lucru este posibil întrucît fiecare transputer are în interior un circuit PLL al cărui oscilator merge pe o frecvență superioară tactului de intrare. Ca o consecință, nu există nici o constrîngere în acest sens: fiecare transputer poate avea propriul său oscilator reglat pe o frecvență strict determinată (vezi fig. 4.3.).

Canalul de comunicație a fost astfel gîndit încît circuitul imprimat și aria ocupată să fie minimă. Soluția care a satisfăcut aceste cerințe a fost legătura serială.

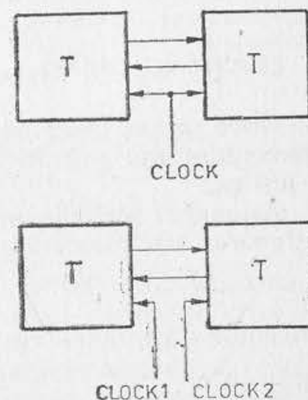


Fig. 4.3. Distribuirea tactului

Protocolul legăturii şi caracteristicile electrice formează un standard pentru toată familia de transputere care cuprinde pe lîngă transputerele de diferite performanţe şi adaptoare de legătură şi transputere de interfaţă. Toate acestea au legături standard care acceptă frecvenţa de transfer de 10 Mbiţi/sec. Semnalele emise şi recepţionate pe aceste legături standard sînt compatibile TTL.

4.3. LIMBAJUL OCCAM

Aşa cum s-a arătat, OCCAM-ul a fost conceput odată cu familia de transputere, ca atare răspunde solicitărilor de concurenţă dar şi celor de configurare şi de exploatare eficientă a resurselor. El poate fi utilizat la programarea unui transputer sau a unei reţele de transputere.

Cînd este folosit un singur transputer, acesta îşi imparte timpul între diferite procese concurente şi comunicaţie; schimburile de date între procese se fac prin intermediul memoriei.

Cînd OCCAM-ul este folosit la programarea unei reţele de transputere fiecare transputer îşi execută procesele alocate; comunicarea între procese se face prin canalele de comunicaţie. Acelaşi program OCCAM poate fi implementat pe diferite configuraţii de reţele de transputere funcţie de anumite cerinţe şi opţiuni în ceea ce priveşte costul sistemului sau timpul de prelucrare a informaţiilor impus.

Limbajul de programare OCCAM lucrează cu procese elementare, cu construcţii şi cu proceduri.

PROCESE ELEMENTARE

Orice proces poate cuprinde alte procese, cu excepţia a 3 procese elementare sau primitive. Aceste procese sînt: asignarea, intrarea şi ieşirea.

Asignarea atribuie unei variabile, V, valoarea unei expresii, e. Asignarea este marcată prin simbolul :=. De exemplu:

x:=x + 1

înseamnă incrementarea valorii lui x sau

sqr:=n*n

înseamnă atribuirea valorii n la pătrat variabilei sqr.

Intrarea atribuie unei variabile x valoarea recepţionată pe canalul c. Intrarea este marcată prin simbolul ?. De exemplu:

c?x

Ieşirea transmite pe canalul c, valoarea expresiei e. Ieşirea este marcată prin simbolul!. De exemplu:

c!e.

Comunicarea între procese de pe transputere diferite se face numai prin canalele standard. Dacă un canal e folosit ca intrare într-un proces el este în mod obligatoriu ieşire dintr-un alt proces. Comunicarea fiind sincronizată, ea are loc cînd ambele procese de intrare respectiv ieşire sînt pregătite.

Procesele pot fi combinate, în funcţie de modul în care se doreşte desfăşurarea lor în raport cu ele însele, în patru moduri: secvenţial, paralel, condiţionat şi alternativ. Procesele subsumate unuia din modurile posibile de desfăşurare amintite pot forma o construcţie. Construcţia în sine este un proces, şi de aceea poate fi folosită de alte procese sau alte construcţii. Pentru a marca apartenenţa unui proces la o construcţie regulile OCCAM cer scrierea acestuia cu 2 spaţii mai la dreapta.

Construcţia secvenţială are următoarea formă:

```
SEQ
  P1
  P2
  P3
```

Procesele componente ale construcţiei secvenţiale, P1, P2 SI P3 sînt executate unul după celălalt în ordinea precizată. Construcţia se termină cînd ultimul proces a luat sfîrşit. Construcţia secvenţială este echivalentă programelor convenţionale, clasice.

Construcţia paralelă are următoarea formă:

```
PAR
  P1
  P2
  P3
```

Aici procesele componente P1, P2, SI P3 se numesc procese concurente şi se execută simultan. Apar două cazuri distincte. Primul cînd e vorba de un singur transputer pe care se execută construcţia paralelă. Situaţie în care la o anumită scară de timp, la care percepţia este de simultaneitate, putem justifica noţiunea de concurenţă. În fond, derularea este tot secvenţială. Al doilea caz este

cel al unei rețele de transputere unde noțiunea de concurență, de simultaneitate capătă accepțiunea reală a termenului. Construcția paralelă se termină odată cu terminarea ultimului (cel mai lung) proces component. De exemplu:

```
PAR
  c1?x
  z: = z + 1
  c3!y
```

este o construcție paralelă formată din trei procese elementare în care valoarea recepționată pe canalul c1 se atribuie variabilei x, valoarea variabilei y este transmisă pe canalul c3 și variabila z este incrementată, toate acțiunile avînd loc simultan. Trebuie remarcat faptul că în cazul unei construcții paralele pe un singur transputer comunicarea între procese se face prin intermediul memoriei transputerului, iar în cazul unei rețele de transputere comunicarea interprocese se face prin legături standard. Din punct de vedere OCCAM programul este identic.

Construcția condițională are următoarea structură:

```
IF
  condiția 1
    P1
  condiția 2
    P2
  ...
  ...
```

și se interpretează astfel: se execută procesul P1 dacă condiția 1 este îndeplinită; altfel, dacă condiția 2 este adevărată se execută P2 ș.a.m.d. pînă cînd se execută un proces care termină construcția. De exemplu:

```
IF
  z = 7
    c1!x
  z<>7
    c1?z
```

construcția înseamnă că se transmite pe canalul c1 valoarea variabilei x numai dacă variabila z este egală cu 7 altfel atribuie variabilei z valoarea recepționată pe canalul c1.

Construcția alternativă are următoarea formă:

```
ALT
  input 1
    P1
  input 2
    P2
  input 3
    P3
```

și se interpretează astfel: se așteaptă pînă cînd unul din procesele de intrare e gata. Dacă comunicația pe canalul i este gata, atunci procesul elementar de intrare input i se va executa, după care se execută și procesul Pi. Numai o intrare împreună cu procesul asociat se execută, după care se termină construcția. Această construcție implementează tratarea evenimentelor externe într-un mod similar întreruperilor cunoscute în cazul microprocesoarelor. De exemplu:

```
ALT
  canal 1 ? front
    NUMĂRĂTOR : = NUMĂRĂTOR +1
  canal 2 ? front
    NUMĂRĂTOR : = NUMĂRĂTOR —1
......
```

înseamnă că fie pe canalul de comunicație canal1 s-a recepționat o valoare care se atribuie variabilei front după care se incrementează variabila NUMĂRĂTOR, fie pe canalul 2 s-a primit o valoare ce se atribuie aceleiași variabile front dar variabila NUMĂRĂTOR se decrementează.

Din punct de vedere logic un program OCCAM, pentru o rețea de transputere sau pentru un transputer nu diferă, după cum și configurația aleasă nu afectează desfășurarea logică a programului.

O construcție paralelă poate fi configurată pentru o rețea de transputere folosind o construcție specială PLACED PAR. Fiecare proces component al construcției PAR va fi executat pe un transputer separat. Variabilele folosite în fiecare proces distribuit trebuie să fi fost declarate anterior pe acel trasputer sau vor fi declarate la începutul procesului în cauză.

OCCAM-ul admite două proceduri: una de repetiție condiționată și una de repetiție determinată.

Repetiția condiționată are următoarea formă:

```
WHILE condition
  P
```

care se interpretează astfel: procesul P se reia atîta timp cît condiția este falsă. De exemplu:

```
WHILE (x/7)<1
  SEQ
    x := x — 2
    c1 ! x
```

ceea ce se traduce prin transmiterea pe canalul c1 a valorii variabilei x diminuată cu 2 atîta timp cît această valoare (a variabilei) este mai mare decît 7.

Repetiția determinată se folosește ca o construcție pentru a relua procesele componente de un număr de ori specificat. De exemplu, pentru a relua un proces P de n ori, adică pentru a implementa o buclă se poate dezvolta următoarea structură:

```
SEQ i = 0 FOR n
  P
```

sau pentru a dezvolta o matrice de n procese concurente vom scrie:

```
PAR i = 0 FOR n
  Pi
```

Această ultimă formă este echivalentul următoarei:

```
PAR
  P1
  P2
  P3
  .
  .
  .
  Pn—1
  Pn
```

Deasemenea se poate subsuma procedurilor și un proces definit și botezat de utilizator, care ulterior se apelează numai prin nume. De exemplu, vom defini procesul de ridicare la cub în felul următor:

```
PROC CUB (INT n)
  CUB: = n * n * n
```

care ulterior poate fi apelat în alte procese sub forma CUB (x).

Fiecare variabilă, expresie sau valoare este de un anumit tip. Tipurile recunoscute sînt cele îndeobște cunoscute: întreg cu semn (INT), octet (byte), variabila logică (booleana) dar și unele tipuri

specifice cum sînt: CHAN of și TIMER și în plus matrici și înregistrări de un anumit tip.

Dezvoltarea de programe pentru rețelele de transputere reclamă multă experiență. Dacă în cazul clasic, secvențial, pe un hard dat se scriu programe mai mult sau mai puțin eficiente, în cazul proceselor concurente, complicate și greu de optimizat, se pare că este mai eficient să se dezvolte aplicația logică, după care să înceapă implementarea ei în programe care să exploateze concurența urmărită și numai în final să se realizeze hard-ul care să corespundă nevoilor programului concurent. Această notă, oarecum inedită, corespunde începutului utilizării concurenței.

În cadrul programării concurente se face referire tot mai mult la noțiunea de comportare logică (Logical Behaviour) care cuprinde dezvoltarea logică a unui program înafara noțiunii de timp real. Cei care au scris aplicații din domeniul automatizărilor, pe procesoarele convenționale, s-au lovit de diferența dintre comportarea logică, statistică și comportarea în timp real, dinamică. În cadrul programării concurente comportarea logică nu este afectată de modul cum sînt distribuite procesele între transputere sau de către viteza de procesare și de comunicare. Pe sistemul de dezvoltare trebuie să ne asigurăm că o execuție diferă de alta numai în termeni bine determinați de datele de intrare.

Deoarece în OCCAM un program este în general o succesiune de procese, construcții și proceduri, care la rîndul lor sînt formate din alte procese, este foarte simplu de adăugat un monitor sau un simulator. Prin adăugarea unor procese care să simuleze mediul exterior putem urmări comportarea logică și într-un mediu extern simulat.

Un program scris în C, Pascal, Fortran sau chiar OCCAM compilat separat poate fi executat numai pe un transputer. Dacă programul este scris în alt limbaj decît OCCAM atunci acest program trebuie prevăzut cu un sistem de desfășurare în timp cît și cu un mod de acomodare la conceptul OCCAM de canale de intrare/ieșire. Astfel de subprograme compilate separat sînt legate de un cadru general OCCAM. Acest cadru OCCAM include informațiile de configurare (și specifică și în ce transputer particular se va executa subprogramul).

4.4. INTERFAȚAREA

Toți membrii familiei de transputere au una sau mai multe legături standard, ceea ce permite interconectarea lor. Pentru interfața cu mediul exterior s-au conceput transputere dedicate (Peripheral

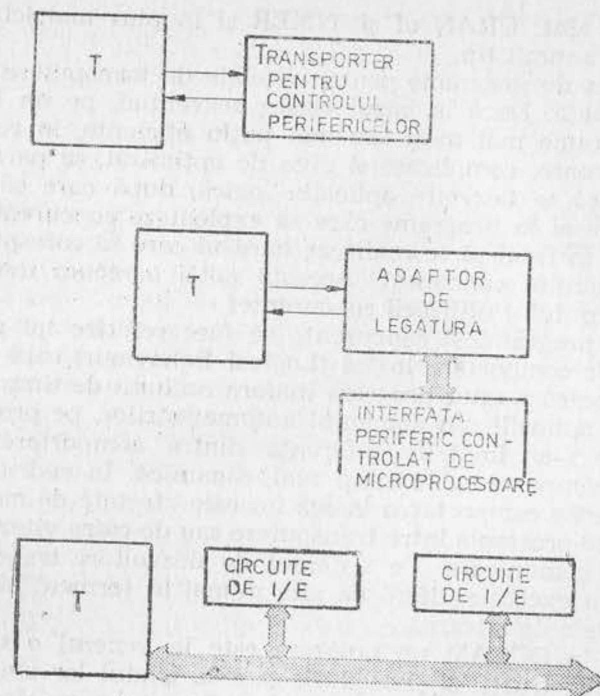


Fig. 4.4. Interfețe cu perifericele

control transputer) unor familii de periferice, cum ar fi ecranul grafic sau discul de memorie și adaptoare de interfață (Link adaptor). Nu vom mai itera la fiecare caz în parte faptul că legăturile serie ale acestora respectă întrutotul specificațiile stabilite pentru transputer-ul propriu-zis și deja prezentate. În fine, mai există o a treia cale de interfațare folosind circuite de intrare/ieșire din familiile microprocesoarelor INTEL, MOSTEK, ZILOG etc. Aceste circuite se includ în spațiul de memorie și vor fi privite de procesor pe a cărui magistrală sînt puse ca niște locații sau zone de memorie. Vor fi accesate, prin urmare, ca atare. Transputerele dedicate au înglobat un hardware special destinat controlului perifericului respectiv. Aplicația software care împreună cu acel hardware dedicat controlează perifericul în cauză este privită ca un proces OCCAM. Proces la care alt transputer din rețea are acces prin legătura standard.

A doua metodă de interfațare folosește tot un circuit integrat, de mai mică complexitate însă, un adaptor de comunicație între

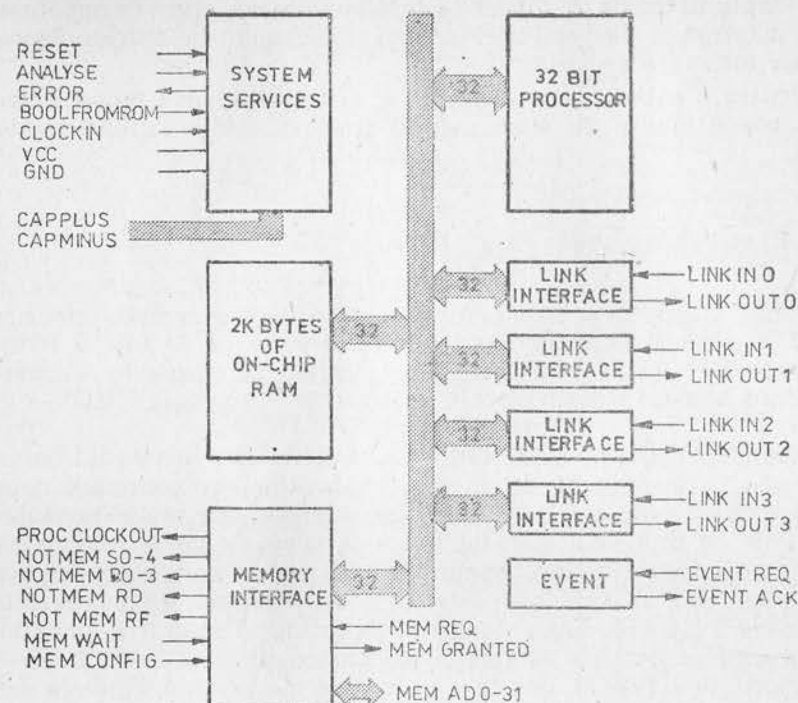


Fig. 4.5. Transputerul T414

legătura standard și o interfață mai generală. În figura 4.6 se prezintă un astfel de adaptor standard al firmei IMMOS, COII. Acest adaptor convertește legătura serială full duplex de la transputer într-o interfață paralelă cu semnale de control (handshake). Mai mult decît atît adaptorul poate fi configurat în două moduri. În modul 1, el convertește legătura serială standard în 2 interfețe paralele unidirecționale cu semnale de control. O interfață pentru datele care vin pe legătura serie de la transputer spre periferic și o interfață pentru datele care vin de la periferic spre transputer.

În cel de-al doilea mod adaptorul face legătura între comunicația standard și o singură interfață bidirecțională de 8 biți; echivalentă unei magistrale microprocesor.

În acest mod se folosesc registre de stare și control și de date atît pentru direcția de ieșire cît și pentru direcția de intrare. În cadrul acestei interfețe există și 2 linii de întrerupere care pot fi mascate sau nu de către doi indicatori dedicați.

Adaptorul poate fi folosit la interconectarea altor transputere, a unor controlere de periferice a unor subsisteme de intrare/ieşire şi a unor microprocesoare.

Pentru a coborî de la concepte şi generalităţi în concret prezentăm în continuare pe scurt primul transputer din familia INMOS, T414.

4.5. TRANSPUTERUL T414

Acest transputer este primul din familia dezvoltată de firma INMOS. IMS T414 integrează un procesor de 32 de biţi, 2 Kocteţi de memorie citeşte/scrie rapidă, o interfaţă de memorie, 4 legături standard într-un singur circuit, realizat în tehnologie CMOS.

PROCESORUL pe 32 de biţi asigură aritmetica în virgula flotantă (microcod). Procedurile de chemare subrutine, de comutare a proceselor concurente şi de servire a întruperilor sînt realizate toate în intervale de timp mai mici de 1 μsec. Procesorul poate împărţi timpul între oricîte procese concurente. Un proces care aşteaptă o comunicaţie sau un ceas nu consumă timp. Pentru rapidizarea întreruperilor T414 are două nivele de priorităţi. Declararea priorităţilor proceselor paralele se face prin construcţia PR1 PAR. Procesele de prioritate 1 (joasă) sînt executate cînd nu există solicitarea de la procesele de prioritate 0 (înaltă). Se declară procese de prioritate înaltă în general acele procese ce au durata scurtă, deoarece procesele de prioritate înaltă monopolizează procesorul. Dacă unul sau mai multe procese de prioritate înaltă sînt gata să înceapă, atunci unul dintre ele va fi selectat şi va rula pînă cînd va intra într-o stare de aşteptare pentru o comunicaţie sau va aştepta o intrare de ceas.

Dacă nu există nici un proces de prioritate 0, dar există mai multe de prioritate joasă atunci unul dintre acestea este selectat. Procesele de prioritate joasă sînt divizate în timp (timesliced). Fiecare interval acordat unui astfel de proces durează 5120 tacţi (CLOCKIN) ceea ce înseamnă aproximativ 1msec pentru o frecvenţă de ceas de intrare de 5MHz. Pentru a fi siguri că procesele de joasă prioritate vor începe, procesele de prioritate 0 nu trebuie niciodată să ocupe procesorul pentru o perioadă de timp egală sau mai mare decît un interval de 1msec (timeslice). De aceea dacă există n procese de prioritate 0 atunci fiecare trebuie să-şi limiteze durata la mai puţin de 1/n dintr-o milisecundă. Procesorul are circuite de ceas necesare proceselor de înaltă şi joasă prioritate.

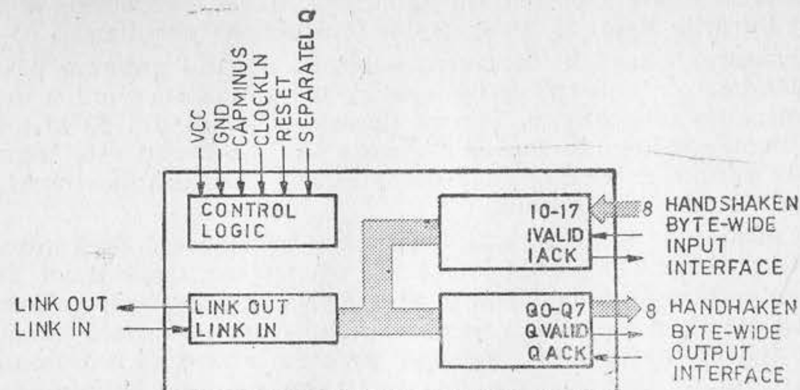


Fig. 4.6. Adaptorul de interfaţă CO11

LEGĂTURILE. Transputerul T414 dispune de 4 legături standard. Pentru transferul de date sau mesaje între memoria sa şi alte transputere se foloseşte transferul de blocuri date de tip DMA. Procesorul propriu-zis şi interfeţele de legătură lucrează independent, prin urmare pot fi privite ca două procese concurente.

MEMORIA. În transputer este înglobată o memorie citeşte/scrie, 2 Kocteţi, cu dublu acces dinspre procesor şi dinspre legături. Memoria este foarte rapidă avînd rata de transfer de 80 Mbiţi/sec. Pentru restul necesarului de memorie s-a înglobat în circuit o interfaţă de memorie pentru un spaţiu de 4 Gocteţi. Datele şi adresele folosesc aceeaşi magistrală multiplexată, magistrala avînd lăţimea de 32 de biţi. Controlerul asociat acestei interfeţe asigură toate semnalele necesare inclusiv cel de refreşare a memoriei dinamice. Schema bloc a transputerului este redată în figura 4.5.

Cuvintele de 32 biţi sînt organizate pe patru octeţi. Mapa memoriei cuprinde, aşa cum am văzut şi în cazul microcalculatorului integrat Z8, şi memoria internă înglobată. Spaţiul de memorie delimitat de adresele 8000.0000H şi 80000.07FFH este ocupat de memoria internă. Primele 18 cuvinte din spaţiul de adresă sînt rezervate pentru nevoile sistemului. Prima locaţie la dispoziţia utilizatorului este 12H, şi se numeşte MemStart.

LANSAREA (Bootstrapping). Transputerul se lansează la punerea sub tensiune sau după un semnal de reset ca oricare procesor, dar tot el mai acceptă o relansare specială care permite examinarea stării sistemului. În cazul cînd e o reţea complexă de transputere

această facilitate e extrem de importană. Această relansare se face dacă intrările Reset și Analyse sînt sensibilizate simultan.

Lansarea constă în începerea execuției codului program păstrat în ROM sau a unui cod recepționat pe o legătură standard și depus în memoria citește/scrie. Pentru lansarea codului din ROM intrarea BootFromRom se ține la valoarea Vcc. Controlul este transferat la adresa #7FFF.FFFEH din memorie care conține invariabil un salt la adresa ROM-ului.

Lansarea de la o legătură standard se inițializează dacă intrarea BootFromRom este ținută la masă. În acest caz, după Reset T414 așteaptă primul octet de control să sosească pe oricare din cele 4 legături standard ale sale. Valoarea acestui octet reprezintă numărul de octeți ce urmează a fi încărcați pe acea intrare, ei reprezentînd programul de lansare. Implicit acest prim caracter trebuie să fie mai mare decît 1. Următorii octeți sînt plasați în memorie de la adresa MemStart. În acest caz adresa MemStart este #8000.0048H, deci se folosește memoria internă care există întotdeauna indiferent de implementarea realizată. După preluarea celor n octeți procesul începe execuția codului depus începînd cu adresa MemStart. Această execuție este un proces de joasă prioritate.

Transputerul T414 permite înscrierea sau citirea oricărei locații de memorie, fie ea internă fie externă, de către și spre legătura standard. Dacă primul caracter care vine, în cazul lansării de pe legătura serială, este 0, atunci urmează obligatoriu un cuvînt de adresă și apoi un cuvînt de date care este înscris la adresa anterior furnizată. Dacă acest prim caracter este 1, atunci urmează un cuvînt de adresă după care cuvîntul de date de la adresa anterior furnizată este citit din memorie și transmis pe lina de legătură. Această facilitate asigură unui program adecvat depanarea ușoară și din mers chiar în sisteme în curs de punere în funcțiune.

BIBLIOGRAFIE

1. BENNEWITZ, WERNER ș.a. — PROGRAMMIERUNG VON EINCHIP MIKRO RECHNERN, BERLIN, 1987
2. * * * — Z8 MICROCOMPUTER. PRELIMINARY TECHNICAL MANUAL
3. * * * — ZILOG. CATALOG 86
4. BLAKESLEE, TH. — PROIECTAREA CU CIRCUITE LOGICE MSI și LSI STANDARD, București, Ed. Tehnică, 1988.
5. * * * — MCS — 48 MICROCOMPUTER USER'MANUAL. INTEL
6. KAFRIESSEN, E. ș.a. — INDUSTRIAL ROBOTS AND ROBOTICS
7. CĂPĂȚÎNĂ, O. ș.a. — PROIECTAREA CU MICROPROCESOARE, Cluj-Napoca, Ed. Dacia, 1983.
8. SĂNDULESCU, GH. — PROTECȚIA LA PERTURBAȚII ÎN ELECTRONICA INDUSTRIALĂ ȘI AUTOMATICĂ, Ed. Tehnică, 1985
9. * * * — PIC 16C5x SERIES MICROCHIP TECHNOLOGICAL
10. * * * — TRANSPUTER REFERENCE MANUAL, INMOS, 1986
11. * * * — M80—UC. MANUAL DE UTILIZARE, MICROELECTRONICA BUCUREȘTI
12. RĂDOI, C. ș.a. — CIRCUITE ȘI ECHIPAMENTE ELECTRONICE INDUSTRIALE, Ed. Tehnică, 1986
13. * * * — ELECTROTEHNICA, ELECTRONICA, AUTOMATICA, București, Anul 32, nr. 3, august 1988
14. KUZNIA, C. — PROCESSOR WITH 128 MICROCOMPUTERS (I 8080), EUROMICROJOURNAL, vol 5, m 1

Societatea Comercială ERS CUG S.A.
3400 Cluj-Napoca, Bd. Muncii nr. 18
telefon 95-142888, telex 31335, fax 095-154746

Execută la cererea agenților economici și a persoanelor fizice, pe bază de comenzi ferme și contracte, următoarele tipuri de lucrări și servicii:

A. În domeniul activităților de engineering și consulting:

— încercări mecanice, analize fizico-chimice și spectrale, în sistem de asigurare a calității;
— verificări și atestări metrologice pentru A.M.C.;
— proiectare de sisteme informatice și informaționale pentru activități de birotică și conducerea producției, precum și programe la temă pentru tehnologii de fabricație pe mașini cu comandă numerică;
— proiectare și execuție sisteme de încălzire, uscare și tratament termic pentru domenii industriale, folosind tehnologia microundelor;
— realizează cursuri de perfecționare în meserii specifice construcțiilor de mașini;
— execută lucrări de traducere și retroversiuni cu specific tehnic în și din limbile: engleză, franceză și rusă;

B. În domeniul activităților de producție:

— repară, recondiționează și modernizează orice fel de mașini unelte așchietoare, prese mecanice și hidraulice, bobinează aparate și mașini electrice pînă la puteri de 1000 kW;
— execută lucrări de antrepriză generală și construcții montaj cu predare la cheie;
— livrează oxigen lichid și gazos, argon și azot la purități înalte;
— închiriază spații de depozitare în magazii de tip stive înalte și containerizat cu acces auto și CFU.

C. În domeniul social și sportiv:

— organizează tabere și cantonamente pentru elevi și sportivi la casa de odihnă proprie de la Mărișel, în serii de 7—10 zile, maxim 40 pers./serie, cu acces la baza sportivă;
— pune la dispoziție baza sportivă din cart. Gheorgheni, care dispune de terenuri de handbal, tenis și fotbal și de instructori specializați.



BANCA „DACIA-FELIX" S.A.
CLUJ-NAPOCA
str. Memorandumului nr. 28
tel. 40.95.11.44.33
telefax 40.95.11.79.36
telex 31.374

Oferă:
— contractarea de credite în lei și în valută din țară și din străinătate
— acordarea de împrumuturi în lei și în valută tuturor categoriilor de agenți economici și persoanelor fizice
— deschiderea de conturi pentru clienți și efectuarea de oparațiuni de încasări și plăți în lei și valută în contul acestora
— emiterea, confirmarea și efectuarea de operațiuni cu scrisori de garanție, avaluri, cauțiuni, în țară și în străinătate
— participarea la consorții de garanții și la credite consorționale interne și internaționale
— participarea la operațiunile de licitație valutară, cumpărarea și vînzarea de metale prețioase, efectuarea de operațiuni de schimb valutar
— orice alte operațiuni și servicii bancare

Sucursale:

— CLUJ-NAPOCA
str. Memorandumului
nr. 28
tel. 095/11.44.33, telex 31. 374
telefax 095/11.54.57
— BRAȘOV
str. Turnului nr. 5
tel. 0921/51321, 63422
— BISTRIȚA
str. Odobescu nr. 17
tel. 0990/22126, 22621
— BAIA MARE
Bd. 22 Decembrie nr. 40
tel. 0994/11742
— ALBA IULIA
Calea Moților nr. 1
tel. 0968/11299, 26979
— ORADEA
Pța Independenței nr. 35
tel. 0991/346492, 39276
— BUCUREȘTI
Șoseaua Ștefan cel Mare
nr. 7—9
tel. 01/6113224
— ZALAU
Piața Libertății nr. 12
tel. 096/32442
— CONSTANȚA
str. N. Titulescu nr. 7
tel. 091/616954
— DEJ
Piața 16 Februarie nr. 4
tel. 095/213592
— TURDA
Pța Libertății nr. 1
tel. 095/316410
— CÎMPIA TURZII
str. Băii nr. 7
tel. 095/368952

INSTITUTUL DE TEHNICĂ DE CALCUL
str. Republicii 109
3400 Cluj-Napoca
tel. 40/95/116060
fax. 40/95/111236

OFERĂ
multe aplicații, pe calculatoare compatibile cu IBM — PC, între care și:

MEDIMAG
Sistem de îmbunătățire și arhivare a imaginilor medicale

Imaginile preluate de la ecograf, angiograf sau aparat RX sînt preluate pentru a fi îmbunătățite calitativ și apoi arhivate sau incluse într-o bază de date medicale despre pacienți.

GESEX
Generator de sisteme expert

Permite elaborarea de sisteme expert cu instrumente evoluate de achiziție a cunoștințelor și de testare a corectitudinii inferențelor efectuate. Produsul este util în domenii pentru care expertiza umană poate fi formalizată într-o bază de cunoștințe, ca de pildă:
— medicină — asistarea diagnosticării
— geologie — prognoza zonelor cu zăcăminte potențiale
— drept — consultanța juridică
— învățămînt — instruire asistată
— agricultură — asistarea diagnosticării în medicină, fitopatologie
— genetică și bioinginerie.

Toate termenele de livrare sînt sub 30 de zile de la înregistrarea comenzii.